IMPLEMENTASI *LOW POWER* PADA *WIRELESS SENSOR NODE* UNTUK AKUISISI DATA LAHAN PERTANIAN DENGAN METODE *WAKE UP INTERRUPT* MODUL RTC

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Aditya Geraldo
135150300111009

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI *LOW POWER* PADA *WIRELESS SENSOR NODE* UNTUK AKUISISI DATA LAHAN PERTANIAN DENGAN METODE *WAKE UP INTERRUPT* MODUL RTC

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Aditya Geraldo
NIM: 135150300111009

Skripsi ini telah diuji dan dinyatakan lulus pada
31 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbingl	Dosen Pembimbing II
Sabrtansyah Rizqika A., S.T, M.Eng	Rakhmadhany Primananda, S.T, M.Kom
NIP: 19820809 201212 1 004	NIK: 2016098604061001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan S.T M.T Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 31 Juli 2018

Aditya Geraldo

NIM: 135150300111009

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah melimpahkan kasih, berkat serta anugrahNya sehingga laporan skripsi yang berjudul "Implementasi *Low Power* Pada *Wireless Sensor Node* Untuk Akuisisi Data Lahan Pertanian Dengan Metode *Wake Up Interrupt* Modul RTC" ini dapat terselesaikan dengan baik.

Penulis menyadari bahwa penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Oleh sebab itu, penulis menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng selaku dosen pembimbing pertama yang telah memberikan support dan bimbingannya kepada penulis untuk segera menyelesaikan skripsi ini.
2. Bapak Rakhmadhany Primananda, S.T, M.Kom selaku dosen pembimbing kedua yang selalu membantu penulis dalam menyelesaikan laporan skripsi.
3. Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika.
4. Kedua orang tua penulis yang terus memberikan semangat serta doa kepada penulis.
5. Ahmad Faris Adhnaufal, Faviansyah Arianda Pallas, Pramudya Mahardika, Riodam Sayyid Abiddin, Farchrur Febriansyah, Fauzi Rivani, Erdano Sedya, Vatikan Aulia Makkah, Arief Indra Rivaldy Permana, Rinaldi Albert, Ikhwan Zulfy dan teman- teman Kontrakan Ceria yang selalu menyemangati dan mensupport penulis dengan berbagai hal dan nasihat sehingga skripsi ini dapat terselesaikan.
6. Teman-teman terbaik di program studi teknik komputer angkatan 2013 yang selalu memberikan semangatnya sehingga penulis dapat menyelesaikan skripsi ini.
7. Dan orang-orang yang selalu mendukung dan mendokan penulis yang tidak dapat penulis ucapakan satu persatu. Terimakasih atas semua doa dan support baik inmateril maupun non meteril.

Penulis mengucapkan banyak terima kasih dan penulis sadar akan adanya beberapa kekurangan pada laporan ini. Jadi penulis berharap adanya penyempurnaan dari pihak-pihak terkait. Semoga laporan ini dapat memberikan manfaat dan referensi untuk melakukan penelitan dalam penyempurnaan sistem.

Malang, 31 Juli 2018

Penulis

Aditya Geraldo

ABSTRAK

Aditya Geraldo, Implementasi Low Power Pada Wireless Sensor Node Untuk Akuisisi Data Lahan Pertanian Dengan Metode Wake Up Interrupt Modul RTC

Pembimbing : Sabriansyah Rizqika Akbar, S.T, M.Eng dan Rakhmadhany Primananda, S.T, M.Kom

Dalam perkembangannya *Wireless Sensor Network* (WSN) memiliki peran yang begitu penting bagi hidup manusia. Kebutuhan akan perlunya suatu sistem untuk melakukan *monitoring* terhadap lingkungan menyebabkan diperlukannya sistem yang mampu bertahan dalam kondisi sumber daya yang terbatas. Oleh karena itu diperlukan mekanisme atau metode yang mampu mengatasi hal tersebut. Pada penelitian sebelumnya telah dibuat cara penghematan daya yang dilakukan secara *software* tapi dalam penelitian ini penulis melakukannya secara *hardware.* Dalam penelitian ini penulis telah berhasil mengintegrasikan suatu metode yang disebut sebagai *wake up interrupt* yang memanfaatkan modul *Real Time Clock* (RTC) untuk menjalankan *mode low power* dalam tiap *node sensor* yang dibuat. *Node sensor* terdiri atas Arduino Pro Mini sebagai modul *processing,* NRF24L01 sebagai modul komunikasi *wireless* serta modul RTC. Dari hasil penelitian yang dibuat, mekanisme penghematan daya ini mampu melakukan penghematan daya hingga 81,25% dari penggunaan arus saat *node sensor* saat telah memasuki *mode sleep.*

Kata kunci : *Wireless sensor network*, RTC, *low power, mode sleep, sensor node*

ABSTRACT

Aditya Geraldo, ***Low Power Implementation In Wireless Sensor Node For Agricultural Data With RTC Module's Wake Up Intterupt Method***

Counselor **: Sabriansyah Rizqika Akbar, S.T, M.Eng dan Rakhmadhany Primananda, S.T, M.Kom**

In the development of Wireless Sensor Network (WSN), it has an important role for human life. The need for a system to monitor the environment leads to the need for system that can survive under resource limited condition. Therefore it is necessary for mechanism or method that can overcome this problem. In the previous research has made a way of power savings made in software but in this study the author has done a hardware implementation instead. In this research, the authors have successfully integrated a method called wake up interrupt that utilizes the Real Time Clock (RTC) module to run low power mode in each sensor node that has been created. This sensor node consists of Arduino Pro Mini as a main processing module, NRF24L01 as a wireless communication module and RTC module that divided into two nodes which are receiver node and transmitter node. From the results of this research, the power saving mechanism is capable of saving power up to 81.25% of used current when the sensor node has entered the sleep mode.

Keywords*: Wireless sensor network, RTC, low power, sleep mode, sensor node*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Secara umum *Wireless Sensor Network* (WSN) dapat dijelaskan sebagai suatu kumpulan *node-node* sensor yang saling terhubung dan berkomunikasi dalam melakukan pengamatan pada suatu lingkungan terkontrol (Bröring, A. et al.,2011). Hal ini memungkinkan adanya interaksi seorang atau komputer dengan lingkungan yang ada di sekitarnya. Dalam dunia WSN pemantauan dilakukan dengan menyebar lebih dari satu unit *wireless sensor node* di dalam suatu area. Sehingga dibutuhkan sumber daya yang memadai agar seluruh *node* dapat bekerja dengan baik. Namun juga terdapat permasalahan lain yaitu untuk saling berkomunikasi dengan baik tiap sensor *node* harus memiliki sistem penjadwalan yang baik.

Penggunaan teknologi WSN dapat diterapkan pada berbagai bidang seperti pertanian, yang sangat membutuhkan pemantauan yang teratur dan efisien. Salah satu permasalahan yang ada adalah bagaimana cara agar setiap *node* sensor yang melakukan pemantauan serta pengiriman data dapat menggunakan energi yang efisien karena konsumsi daya secara terus menerus dapat menghabiskan baterai (Adhnaufal,2017).

Pada penelitian sebelumnya ada yang telah melakukan penghematan sumber daya dan penjadwalan secara *software*. Salah satunya adalah laporan skripsi yang berjudul "*IMPLEMENTASI LOW POWER WIRELESS SENSOR NETWORK UNTUK PENGUKURAN SUHU BERBASIS NRF DENGAN PENJADWALAN PENGIRIMAN DATA"* yang dilakukan oleh Ahmad Faris Adhnaufal. Dalam penelitian tersebut telah menggunakan *library* khusus yang bernama *Jeelib Library* pada arduino serta penggunaa *Timing-Sync Protocol for Sensor Network* (TPSN) untuk sinkronisasi waktunya.

Dalam penelitian ini, peneliti akan melakukan penghematan daya secara *hardware* dengan menambahkan fitur pada penelitian sebelumnya. Fitur tersebut adalah penjadwalan waktu pengiriman dengan menambahkan modul *Real Time Clock* (RTC) untuk sinkronisasi waktu yang lebih presisi dan dengan adanya penambahan metode *wake up interrupt*. Pada metode ini, penulis akan memanfaatkan salah satu fitur pada modul RTC yaitu alarm. Alarm ini berfungsi untuk mematikan (*low power mode*) atau membangunkan *node-node* yang digunakan pada penelitian ini. Selain itu dengan adanya penambahan tiga buah sensor seperti sensor LDR, sensor *rain drop* dan sensor *soil moisture*.

Berdasarkan latar belakang tersebut, penulis mengharapkan agar penambahan fitur-fitur diatas dapat membuat optimisasi *low power* pada *wireless sensor node* dapat lebih maksimal dan juga manfaatnya dalam akuisisi data dalam bidang pertanian. Implementasi dilakukan menggunakan perangkat Arduino Pro

Mini dengan NRF sebagai modul *wirelessnya* serta penggunaan sensor-sensor untuk lahan pertanian.

1.2 Rumusan Masalah

Berdasarkan latar belakang diatas, penulis merumuskan beberapa masalah yaitu sebagai berikut :

1. Bagaimana mengimplementasikan fitur penjadwalan yang didukung modul RTC ?
2. Seberapa efisienkah besaran arus saat sistem memasuki *state low power* ?
3. Bagaimana hasil dari *low power mode* yang diterapkan dengan *wake up intterupt* ?

1.3 Tujuan

Berdasarkan rumusan masalah yang telah disebutkan, tujuan dari penulisan skripsi ini adalah sebagai berikut :

1. Mengimplementasikan fitur penjadwalan yang didukung oleh modul RTC.
2. Menerapkan *low power mode* dengan metode *wake up interrupt.*
3. Menghasilkan besaran arus seminimal mungkin saat memasuki *state low power.*

1.4 Manfaat

Penelitian ini diharapkan dapat bermanfaat untuk berbagai pihak. Manfaat dari penelitian ini adalah sebagai berikut:

Bagi Penulis :

1. Sistem ini diharapkan dapat mengingatkan pengguna untuk memonitoring keadaan lingkungan pertanian.
2. Dapat membantu melakukan monitoring dengan periode waktu tertentu secara terjadwal dan akurat.
3. Mampu mengimplementasikan fitur *low power* pada *wireless sensor node* dengan metode *wake up intterrupt*.

Bagi Pembaca :

1. Mengetahui cara membuat *hardware* untuk akuisisi data pertanian yang dikendalikan oleh Mikrokontroler Arduino.
2. Mengetahui salah metode untuk menghemat daya penggunaan *wireless sensor node* untuk sistem monitoring pertanian.

1.5 Batasan Masalah

Agar permasalahan yang dirumuskan lebih terfokus dan mendapatkan hasil sesuai dengan yang diharapkan, maka penelitian ini dibatasi dalam :

1. Sistem berupa *sensor node* beserta *Receiver*.
2. Menggunakan modul Real Time Clock untuk *wake up interrupt*.
3. Pengujian dilakukan hanya pada *node transmitter* yang berjumlah 4 unit serta *node* receiver berjumlah 1 unit.
4. Pengujian terfokus pada pengukuran arus saat *node Low Power* dan *Non-Low Power mode* pada 4 *node transmitter* yang ada saat melakukan fungsinya.

1.6 Sistematika pembahasan

BAB I Pendahuluan

Pada bab ini berisi mengenai latar belakan pembuatan laporan skripsi, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

BAB II Landasan Kepustakaan

Menguraikan tentang landasan teori yang terkait dengan penelitian mengenai sistem yang akan diimplementasikan.

BAB III Metode Penelitian

Menguraikan dan membahas langkah kerja yang dilakukan dalam penulisan skripsi yang tediri dari studi literatur, implementasi, perancangan, analisis kebutuhan dan pengujian.

BAB IV Rekayasa Kebutuhan

Bab ini menjelaskan secara rinci yang terkait meliputi deskripsi umum dari sistem, rekayasa kebutuhan antar-muka sistem, kebutuhan perangkat keras dan lunak, kebutuhan fungsional, kebutuhan komunikasi, kebutuhan performansi, batasan desain sistem dan alur kerja sistem.

BAB V Perancangan dan Implementasi

Bab ini menjelaskan perancangan sistem penelitian sertas implementasi sistem penelitian berupa implementasi pernagkat keras dan perangkat lunak sebagai *output* dari sistem.

BAB VI Pengujian

Bab ini membahas skenario pengujian, hasil serta analisis hasil pengujian.

BAB VII Penutup

Bab ini membahas kesimpulan serta saran yang diperoleh dari rumusan masalah penelitian dengan melakukan analisis dan pengujian.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Tinjauan Pustaka

Dasar atau acuan yang berupa teori-teori atau hasil penelitian-penelitian sebelumnya merupakan hal yang sangat diperlukan untuk dijadikan data dan teori pendukung pada penelitian ini. Salah satu data pendukung terpenting pada penelitian kali ini adalah laporan dari hasil penelitian-penelitian terdahulu yang relevan dengan permasalahan yang dibahas pada penelitian kali ini. Dalam hal ini, penelitian terdahulu yang difokuskan sebagai data acuan adalah yang berkaitan dengan penghematan energi dan penjadwalan pengiriman data dengan waktu yang tersinkronisasi. Oleh karena itu, peneliti melakukan proses pengkajian terhadap beberapa hasil penelitian sebelumnya berupa tesis maupun jurnal-jurnal melalui internet dan studi literatur.

Berdasarkan penelitian sebelumnya yang menjadi dasar pembuatan penelitian ini adalah skripsi yang berjudul "*Implementasi Low Power Wireless Sensor Network Untuk Pengukuran Suhu Berbasis NRF Dengan Penjadwalan Pengiriman Data*" (Adhnaufal, 2017). Menggunakan kombinasi protokol Time Division Multiple Access (TDMA) dan *Timing-Sync Protocol for Sensor Network* (TPSN) yang diterapkan pada *node* yang berjumlah lebih dari satu buah berhasil pada penelitian tersebut. Sinkronisasi waktu antar *node transmitter* dan *receiver* berjumlah rata-rata 30 detik. Sedangkan slot waktu yang digunakan pada metode TDMA yang telah ditentukan sebesar 5 detik. Pada penelitian tersebut telah menggunakan *library jeelib* yang dikombinasikan dengan TDMA dan TPSN sudah berhasil memaksimalkan penggunaan *low power mode* pada *wireless sensor node*nya.

Pada penelitian ini,penulis akan fokus pada penambahan fitur pada penelitian sebelumnya. Dimana pada penelitian tersebut belum terdapat penggunaan *Real Time Clock* (RTC) yang berfungsi sebagai modul sinkronisasi waktu yang lebih akurat, serta untuk melakukan *wake up interrrupt* pada tiap *node*. Berikut merupakan perbedaan dan persamaan antara penelitian sebelumnya dan yang akan dilakukan.

Tabel 2.1 Hasil Penelitian Terdahulu

No.	Tahun	Nama Peneliti	Judul Penelitian	Persamaan	Perbedaan	
					Penelitian Terdahulu	Rencana Penelitian
1.	2016	Ardy Novian Erwanda	*Implementasi Time Synchronization Pada WSN Untuk Metode TDMA*	Penerapan Metode *Timing-sync Protocol for Sensor Network*	Pengujian diterapkan pada *Node* dengan jumlah sedikit (1 buah *node*	Pengujian akan diterapkan dengan jumlah *node* yang lebih

			Menggunakan Algoritma TPSN	(TPSN) pada WSN	*transmitter* & 1 buah *node receiver*)	banyak dan menerapkan *low power mode* dengan metode *wake up interrupt*
2.	2017	Ahmad Faris Adhnaufal	*Implementasi Low Power Wireless Sensor Network Untuk Pengukuran Suhu Berbasis NRF Dengan Penjadwalan Pengiriman Data*	Penerapan pengaturan waktu untuk melakukan penghematan daya pada WSN dan Penjadwalan	Pengujian difokuskan pada pengaturan waktu untuk penghematan daya dan pengukuran suhu ruangan	Penambahan RTC untuk metode *wake up interrrupt* serta penambahan parameter pengukuran cahaya, hujan dan kelembaban tanah

2.2 Dasar Teori

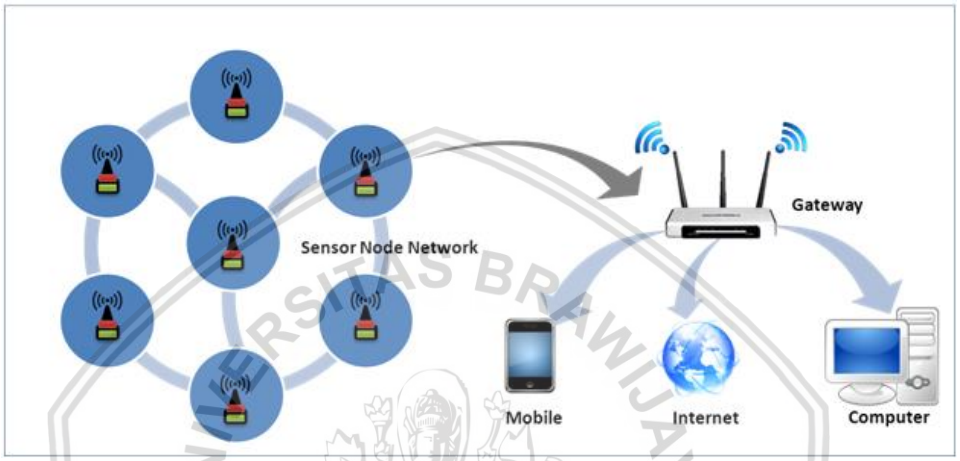
2.2.1 Dasar *Wireless Sensor Network*

Arsitektur WSN tradisional umumnya mengadaptas *flat structure* (yaitu *single-layer planar structure*),juga disebut jenis flat WSN . Sejumlah besar *node* sensor dengan struktur hardware yang sama, dan penginderaan yang minim, pengolahan, dan kemampuan berkomunikasi dikembangkan di daerah pemantauan, dan mengirimkan dan meneruskan informasi yang dikumpulkan oleh *node* sensor lain ke sink *node* menggunakan bentuk multi-hop dengan bantuan *node* lain dalam WSN(Gatut Prasojo,2016). Kemudian WSN yang terhubung dengan jaringan yang lain dengan sink *node*, yang pada akhirnya pengguna dapat mengakses dari jarak jauh, query dan mengelola WSN . Sebagian besar penelitian WSN bertujuan untuk flat WSN, seperti satu-hop jaringan sensor nirkabel , multi tradisional-hop jaringan sensor nirkabel, dan sebagainya.

2.2.1.1 Arsitektur *Wireless Sensor Network*

WSN dibangun dari *access point* dan *sensor point.* Keseluruhan komponen pendukung *WSN* ini akan saling bekerja untuk menyajikan data perubahan karakterisitik dari *Sensor point* dapat juga dikatakan sebagai pemroses data *analog* dari perubahan sensor. Data *analog* sensor akan diubah menjadi *digital* untuk kemudian dikirim ke *access point* melalui media *Wireless* (Bröring, A. et al.,2011).

Access point bertugas untuk melakukan pengontrolan sensor dan menampilkan data sensor dari *access point node*. *Acces point* dapat berupa komputer, perangkat *mobile*, dan lain lain. Salah satu karakteristik khusus yang dimiliki suatu jaringan sensor adalah jumlahnya yang selalu dalam jumlah besar dan kepadatan yang cukup tinggi. Pada suatu kasus monitoring lingkungan, sensor yang digunakan jumlahnya dapat mencapai ratusan sensor yang tersebar secara acak dengan kepadatan tinggi akan tetapi tetap memperhatikan konektivitas, efisiensi dan akurasi. Gambar dari contoh arsitektur WSN dapat dilihat pada Gambar 2.1.

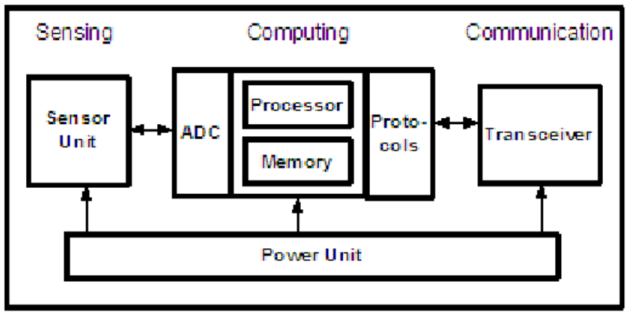


Gambar 2.1 Arsitektur WSN

Sumber : (http://www.rfwireless-world.com)

2.2.1.2 *Sensor Node*

Node sensor adalah salah satu bagian utama dari WSN. Perangkat keras dari *node* sensor umumnya terdiri atas empat bagian: modul daya dan manajemen daya, sensor, mikrokontroler, dan nirkabel transceiver. Modul daya memberikan daya yang dibutuhkan oleh sistem. Sensor adalah modul yang berfungsi untuk mengambil data dari lingkungan maupun alat. Sebuah sensor bertugas mengumpulkan dan mentransformasikan sinyal, seperti cahaya, getaran, dan bahan kimia, menjadi sinyal listrik dan kemudian mentransfer data ke mikrokontroler. Mikrokontroler menerima data dari sensor dan proses data yang sesuai. Transceiver Nirkabel (RF modul) kemudian transfer data, sehingga fisik realisasi komunikasi dapat terjadi dengan baik. Struktur sensor *node* dapat dilihat pada Gambar 2.2.



Gambar 2.2 Struktur Sensor *Node*

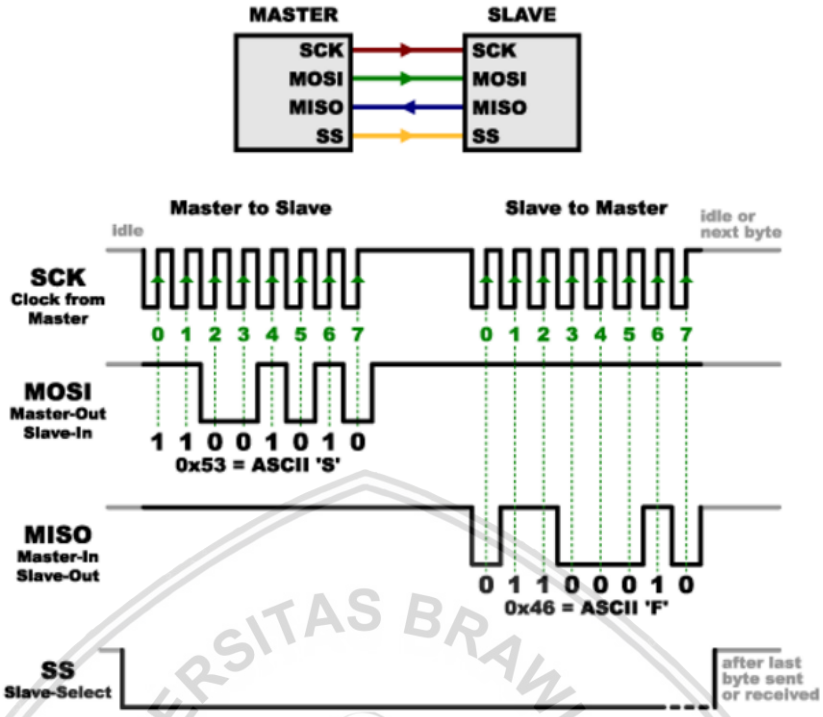
Sumber : (researchgate.net)

2.2.2 Serial Peripheral Interface (SPI)

Serial Peripheral Interface (SPI) merupakan antarmuka komunikasi serial yang tersinkronisasi. SPI digunakan secara spesifik untuk komunikasi jarak pendek yang biasanya digunakan pada *Embedded System*. Perangkat SPI berkomunikasi dengan mode *Full Duplex* dengan menggunakan arsitektur *Slave-Master* yang memiliki satu *Master*. Perangkat *Master* digunakan untuk melakukan pembacaan maupun penulisan. Jika memiliki lebih dari satu perangkat *Slave* maka dapat menggunakan mode pemilihan *Individual Slave Select* (Davis, 2013).

Secara umum, bus SPI memiliki 4 *Logic Signals* yaitu *Serial Clock* (SCLK), *Master Output Slave Input* (MOSI), *Master Input Slave Output* (MISO) dan *Slave Select* (SS). Bus SPI dapat beroperasi dengan sebuah perangkat yang menjadi *Master* dan satu atau lebih perangkat *Slave*.

Prinsip kerja dari pengiriman data SPI menggunakan metode *Master/Slave*. Dimana perangkat yang menjadi *Master* adalah yang biasanya merupakan perangkat yang dapat menghasilkan SCLK dan perangkat yang berfungsi membaca dan menulis data. Ketika terdapat data yang akan dikirim dari *Master* menuju *Slave* maka data akan dikirm melalui jalur MOSI. Jika dari data yang dikirim tersebut membutuhkan respon balik dari *Slave* maka respon tersebut akan dikirim melalui jalur MISO. Pertukaran data tersebut harus tersinkronisasi dengan waktu yang dibuat oleh SCLK sehingga pengiriman transaksi data berjalan secara tersinkronisasi. Ketika terdapat lebih dari satu perangkat *Slave* maka SS akan berfungsi sebagai *Switch* yang digunakan untuk memilih *Slave* yang akan dilayani. Alur prinsip kerja metode *Master/Slave* dapat dilihat pada Gambar 2.11.



Gambar 2.3 Prinsip Kerja Metode *Master/Slave* pada bus SP

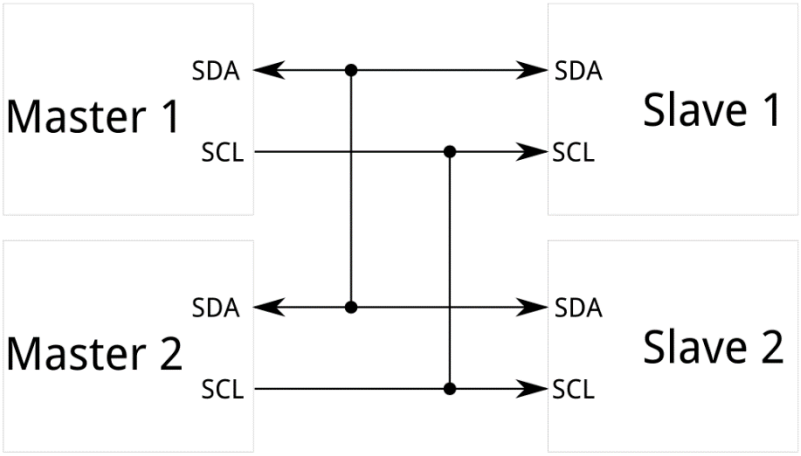
Sumber : (www.learn.sparkfun.com, 2016)

2.2.3 Inter-Intergrated Circuit (I2C)

I2C merupakan suatu protokol komunikasi yang memungkinkan sejumlah *slave chips* untuk berkomunikasi dengan satu atau lebih *master chips.* Seperti pada SPI, I2C hanya untuk komunikasi dalam jarak dekat pada suatu *device.* Pada protokol ini hanya diperlukan dua sinyal untuk saling bertukar informasi.

I2C mampu mendukung hingga 1008 *slave devices.* Namun tidak seperti SPI, I2C mendukung sistem dengan *multi-master* yang mampu berkomunikasi dengan semua perangkat pada suatu *bus*. Namun, setiap *master* tidak mampu saling berkomunikasi satu dengan yang lain, melainkan harus melakukannya secara bergantian.

Hampir semua perangkat yang mendukung I2C mampu berkomunikasi pada *data rate* 100kHz atau 400kHz. Pada tiap 8 bit data yang dikirim akan ditambahkan 1 bit *meta data* yang berisikan "ACK/NACK" bit. Kebutuhan untuk implementasinya pada perangkat keras lebih rumit dibandingkan SPI. Prinsip kerja dari I2C dapat dilihat pada Gambar 2.4.

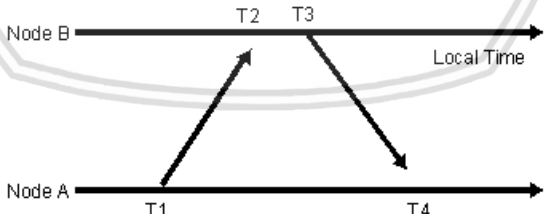


Gambar 2.4 Prinsip Kerja I2C

Sumber : (https://learn.sparkfun.com)

2.2.4 Timing-Sync Protocol for Sensor Network (TPSN)

Time-Sync Protocol for Sensor Networks (TPSN) adalah sebuah algoritma sinkronisasi waktu pada WSN, yang menggunakan pemodelan topologi tree. Dimana terdapat sebuah *node* sebagai *root* yang berada di pangkal tree, sinkronisasi waktu dimulai dari *root* tersebut. Dengan menggunakan aliran data yang disebut *timestamp* yang dikirim dari *root* ke hierarki dibawahnya, waktu disamakan melalui *timestamp* tersebut. TPSN memiliki keunggulan dibandingkan beberapa algoritma lain seperti skala yang lebih besar daripada algoritma RBS (Elson., et al, 2001).



Gambar 2.5 Mekanisme TPSN

Sumber : (Roche, 2016)

Dalam algoritma TPSN terdapat dua fase utama yaitu fase discovery dan fase synchronize. Pada fase discovery *node* yang ada akan membentuk sebuah hierarki untuk pertama kalinya. Dimulai dari *node root* hingga berlanjut ke *node* level dibawahnya. Seluruh *node* dibawah level *root* akan mengingat posisi level nya masing masing serta alamat parent yang menjadi *node root*. Pesan discovery akan terus dikirim dari *root* hingga *node* paling ujung dalam hierarki hingga semua *node* terdaftar. Setelah fase discovery selesai dilanjutkan dengan fase kedua yaitu fase

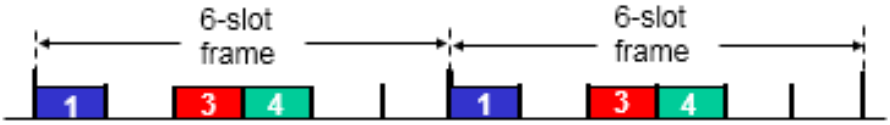
synchronize. Pada fase ini semua *node* akan mulai menyetarakan waktu masing masing hingga memiliki waktu yang sama dengan *node root*. Diawali dari *node* dengan level paling kecil *node* level 1 akan mengirim paket permintaan penyetaraan waktu t1 ke *node root* di level 0. *Root* akan menerima paket permintaan tersebut pada waktu t2. Lalu paket tersebut akan dibalas dengan paket ack yang berisi nomor hierarki t1 t2 dan t3. Kemudian *node* level 1 tersebut akan melakukan perhitungan waktu dari aliran skenario ini dengan melakukan pencocokan waktu. Hal serupa akan dilakukan oleh *node* level 2 ke *node* level 1 dan terus berulang sampai *node* level n pada ujung hierarki. Mekanisme TPSN dapat dilihat pada Gambar 2.5.

2.2.5 Time Division Multiple Access (TDMA)

Protokol *Multi Access* memiliki tiga mekanisme utama, yaitu *channel partitioning, random access,* dan *taking turns*. *Channel Partitioning* adalah protokol yang menggunakan algoritma terdistribusi yang mengatur pembagian *channel* atau *link* yang akan digunakan secara bersama menjadi beberapa potongan yang selanjutnya akan dialokasikan kepada semua *node*. Pada protokol ini terdapat beberapa metode yaitu *Frequency Division Multiple Access* (FDMA), *Time Division Multiple Access* (TDMA) dan *Code Divison Multiple Access* (CDMA).

Taking Turn adalah protokol *Multiple Access* yang bekerja dengan mengatur pengiriman tiap *node* berdasarkan *token polling* yang dikirimkan oleh sebuah master *node* sebagai ”undangan” untuk mengirimkan datanya. Kemudian *node* penerima akan menerima undangan tersebut lalu mengirimkan datanya. Sehingga tidak terjadi pengiriman dalam waktu yang bersamaan. Namun metode ini memiliki beberapa kelemahan seperti *polling overhead, latency* pada pengiriman dan *single point of failure*. (Kurose, et al, 2012).

Time Division Multiple Access atau TDMA adalah metode pembagian akses pada pengiriman data digital yang dibagi berdasarkan teknik pembagian waktu secara multipleksi. Pada dasarnya TDMA bekerja dengan cara membagi frekuensi radio berdasarkan waktu hingga menjadi beberapa slot waktu yang dapat dialokasikan dan mendukung aliran kanal data secara bersama-sama. Transmisi dilakukan dalam bentuk beberapa urutan frame, dimana tiap-tiap frame dibagi menjadi beberapa slot waktu. Dan pada setiap slot waktu bersifat tetap untuk sebuah *transmitter* tertentu. (Stallings, 2005). Setiap *transmitter* akan memiliki sebuah jadwal pengiriman yang berbeda-beda dan memiliki waktu yang tersinkronisasi. Seperti pada Gambar 2.6, terdapat 6 slot waktu pada sebuah frame, dimana beberapa slot digunakan oleh *transmitter* 1, 3 dan 4. Sedangkan slot 2, 5 dan 6 berada pada kondisi idle. Pengiriman akan terus berulang pada frame yang lainnya.



Gambar 2.6 Konsep TDMA

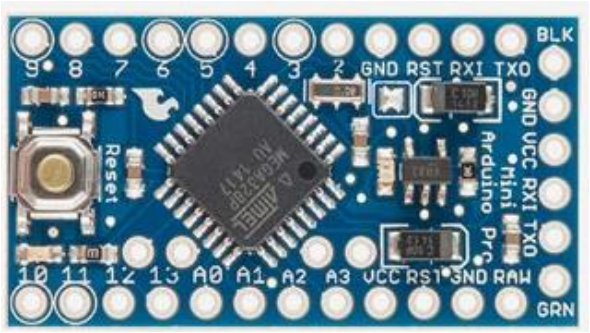
Sumber : (Kurose, et al, 2012)

2.2.6 Arduino

Mikrokontroler adalah sebuah sistem komputer fungsional dalam sebuah chip. Di dalamnya terkandung sebuah inti prosesor, memori (sejumlah kecil RAM, memori program, atau keduanya), dan perlengkapan *input output*. Dengan kata lain, mikrokontroler adalah suatu alat elektronika digital yang mempunyai masukan dan keluaran serta kendali dengan program yang bisa ditulis dan dihapus dengan cara khusus, cara kerja mikrokontroler sebenarnya membaca dan menulis data (Kadir, 2013). Mikrokontroler yang digunakan pada penelitian ini ialah Arduino Uno yang merupakan platform yang terdiri dari *hardware* dan *software*. *Hardware*nya berupa mikrokontroler yang ditambahkan penamaan pin agar mudah diingat. *Software* Arduino merupakan *software open source* sehingga dapat diunduh secara gratis. Software ini digunakan untuk membuat dan memasukkan program ke dalam Arduino. *Software* Arduino terdiri dari 3 bagian, yaitu :

1. *Editor*, untuk menulis dan memanipulasi program (*source code*). Kode program pada Arduino disebut dengan *sketch*.
2. *Compiler*, modul yang berfungsi mengubah *source code* menjadi bentuk biner agar bisa dieksekusi oleh mikrokontroler.
3. *Uploader*, modul yang berfungsi memasukkan kode biner ke dalam memori mikrokontroler.

Dari sekian banyak jenis mikrokontroler Arduino, Arduino Pro Mini adalah salah satu mikrokontroler yang popular saat ini. Dengan ukuran dimensi yang kecil dan kompak membuat mikrokontroler ini terlihat sangat praktis (Arduino.cc,2017). Pada boardnya menggunakan daya 5V dan memiliki *bootloader* dengan frekuensi Kristal 16Mhz. mikrokontroler ini mempunyai beberapa pin digital dan analog serta dilengkapi dengan ADC (*Analog to Digital Converter*). Pemrograman Arduino Pro mini dapat menggunakan kabel serial FTDI yang dihubungkan pada USB komputer dan *software* IDE yang dimiliki oleh Arduino. Berikut Gambar 2.7 menunjukkkan bentuk fisik dari Arduino Pro mini.



Gambar 2.7 Arduino Pro Mini

Sumber : (http://www. Arduino.cc)

Spesifikasi yang dimiliki Arduino Pro Mini dijelaskan pada Tabel 2.2.

Tabel 2.2 Spesifikasi Arduino Pro Mini

Spec/Name	Pro Mini ATmega 168	Pro Mini ATmega 168
Processor	ATmega 168	ATmega328P
Operating/Input Voltage	3.3 V / 3.35-12 V	5 V / 5-12 V
CPU Speed	8 MHz	16 MHz
Analog In/Out	6/0	
Digital IO/PWM	14/6	
EEPROM (kB)	0.512	1
SRAM (kB)	1	2
Flash (kB)	16	32
USB	-	
UART	1	

2.2.7 Sensor Suhu dan Kelembapan

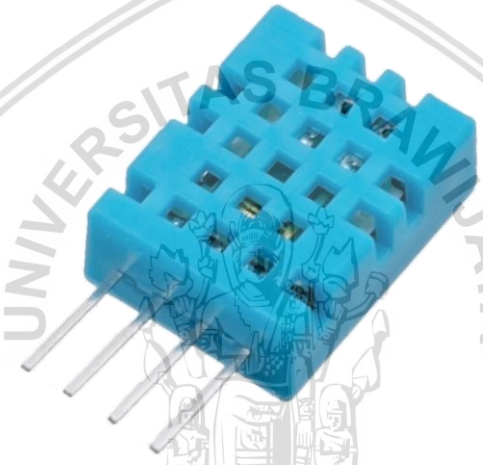
DHT11 merupakan sensor yang dapat mengukur dua parameter lingkungan, yaitu suhu dan kelembapan. Sensor DHT11 memiliki stabilitas yang sangat baik serta fitur kalibrasi yang sangat kuat. Besaran suhu diukur dalam besaran Celcius (°C) dan kelembaban dalam satuan persen (%). DHT11 sering digunakan untuk keperluan monitoring. Dalam sensor ini terdapat sebuah thermistor tipe NTC (Negative Temprature Coefficient) untuk mengukur suhu, sebuah sensor kelembaban tipe resistif dan sebuah mikrokontroler 8-bit yang mengolah hasil sensing dari kedua sensor tersebut dan mengirimnya ke pin output dengan format

single-wire bi-directional (Tian Long, 2011). Spesifikasi dan bentuk fisik sensor DHT11 dapat dilihat pada Tabel 2. 3 dan Gambar 2.8.

Tabel 2.3 Spesifikasi DHT11

Voltage	3~5 Volt
Temperature range	0-50 °C Error ± 2 °C
Humidity	20-90% RH Error ± 5% RH

Sumber : (www.geraicerdas.com, 2016)

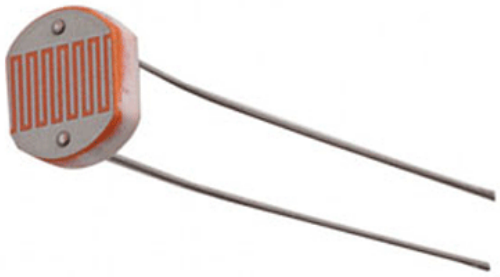


Gambar 2.8 Sensor DHT11

Sumber : (http://irishelectron ics.ie)

2.2.8 Sensor Cahaya

Sensor LDR (*Light Dependent Resistor*) merupakan suatu resistor yang memiliki tingkat sensitivitas terhadap cahaya. Sensor ini dapat membedakan ada atau tidak adanya cahaya serta mengukur intensitasnya. Resistansi pada sensor ini akan sangat besar jika tidak ada cahaya (hingga 1 MΩ), namun jika terpapar oleh cahaya, maka resistansinya akan menurun secara drastis (Xiao-Yuan,2012). Spesifikasi dari sensor ini dapat dilihat dari Tabel 2.4.



Gambar 2.9 Sensor LDR

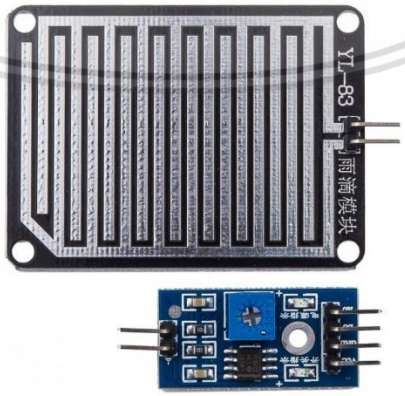
Sumber : (http://resistorguide.com)

Tabel 2.4 Spesifikasi LDR

Voltage AC/DC Peak	320 Volt
Cell Resistance	400 Ohm – 9 K Ohm
Current	75 mAmpere Max
Operating Temprature	-65^{o} – 75^{o} Celcius

2.2.9 Sensor Rain Drop

Sensor Rain Drop merupakan modul sensor yang dapat mendeteksi adanya tetes air hujan serta intensitasnya. Pada modul Ini terdari panel sensor yang merupakan *circuit board* LM353 yang berfungsi sebagai pengukur curah hujan berdasarkan jumlah tetesan air yang jatuh pada *board* tersebut (Ishikawa,2004) . Spesifikasi dari sensor ini dapat dilihat pada Tabel 2.5.



Gambar 2.10 Sensor Rain Drop

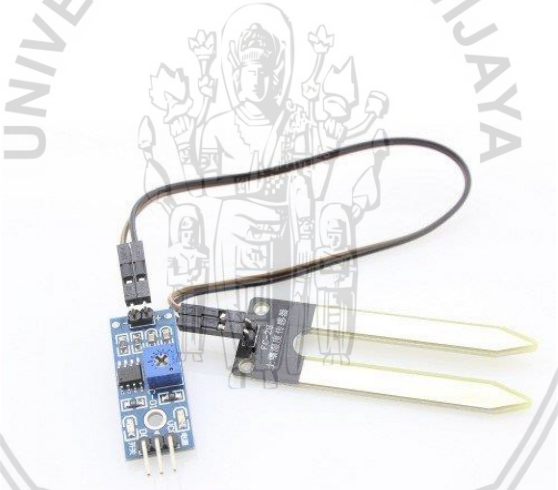
Sumber : (http:// elabpeers.com)

Tabel 2.5 Spesifikasi Sensor Rain Drop

Minimum wet area	0.05 cm^2
Sensing area	7.2 cm^2
Supply voltage	12 VDC ± 10 %
Supply current	150 mA
Operating Temprature	-15-55 °C

2.2.10 Sensor Soil Moisture

Sensor *soil moisture* merupakan sensor yang berfungsi untuk kelembaban pada permukaan tanah (Pelletier,2006). Cara kerja sensor ini adalah dengan cara mengukur jumah kadar air yang ada pada suatu permukaan tanah. Jika tanah memiliki kadar air yang tinggi, maka nilai *output* akan semakin besar, sebaliknya jika semakin rendah maka *output* semakin kecil. Spesifikasi dapat dilihat pada Tabel 2.6.



Gambar 2.11 Sensor Soil Moisture

Sumber : (https : //circuit.rocks)

Tabel 2.6 Spesifikasi Sensor Soil Moisture

Voltage	3.3 – 5 V
Current	35 mA
Output value	0 - 300 (Dry)
	300 - 700 (Humid)
	700 – 950 (Water)

2.2.11 Modul NRF24L01

NRF24L01 merupakan chip wireless yang bekerja pada gelombang 2,4 – 2,5 GHz. Chip ini dapat bekerja sebagai pemancar (Tx) dan penerima (Rx) dan bekerja pada kondisi low-power, sehingga membuatnya dapat bertahan lama saat digunakan untuk mengamati keadaan lingkungan (). NRF24L01 memiliki 8 buah pin yaitu VCC (3.3V DC), GND, CE, CSN, MOSI, MISO, SCK, IRQ. Dalam proses komunikasinya nRF24l01 menggunakan antarmuka SPI dengan rate 0~8 Mbps. Modul ini juga terdapat tipe dengan PA (Power Amplifier) dan LNA (Low Noise Amplifier) sehingga jarak transfer data dapat semakin jauh dan lebih stabil. Area yang dapat dijangkau oleh modul ini mencapai radius 1000m pada lapangan terbuka. Bentuk fisik dari nrf24l01 dapat dilihat pada Gambar 2.12.

Gambar 2.12 NRF24L01

Sumber : (http://img.dxcdn.com/)

Spesifikasi lain yang dimiliki oleh modul ini sebagaimana akan dijelaskan pada Tabel 2.7.

Tabel 2.7 Spesifikasi NRF24L01

Power Supply	1.9V~3.6V
I/O port Working Current	13.5mA at 2 Mbps
I/O port Working Current	0~3.3V, 5V
Data Rate	256kbps/ 1Mbps/ 2Mbps
Receiving Sensitivity	-85dBm at 1Mbps
Transmission Range	70~100 meter at 256kbps
Temperatures Range	Operating:-40^{0}C~85^{0}C Storage :-40^{0}C~125^{0}C

Sumber : (www.nordicsemi.com, 2016)

2.2.12 Real Time Clock (RTC)

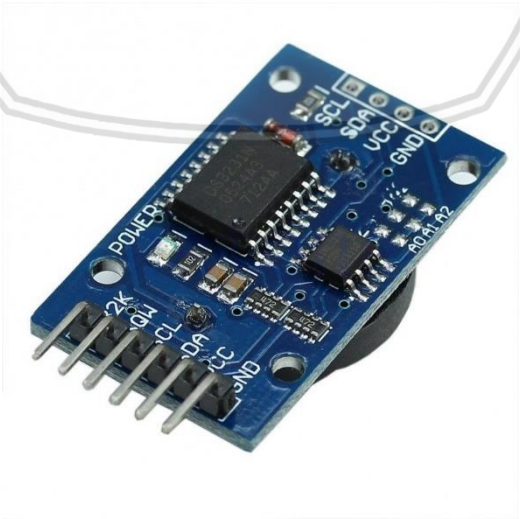
Real Time Clock merupakan suatu chip yang berguna untuk mengitung waktu mulai dari detik, menit, jam, tanggal , bulan dan tahun. Chip ini dapat menghitung waktu secara akurat serta menyimpannya secara *real time.* Dalam mikrokontroler biasanya memiliki fungsi millis () yang berfungsi untuk menampilkan lama waktu standby dari mikrokontroler tersebut (Chapin, 2009). Namun , mikrokontroler tidak dapat mengenal format jam dan tanggal seperti pada umumnya karena fungsi millis hanya mencatat waktu sejak mikrokontroler diaktifkan. RTC berguna untuk melakukan penghitungan waktu yang lebih akurat pada mikrokontroler. RTC pada penelitian ini berfungsi sebagai modul utama untuk melakukan *wake up interrupt*. Bentuk fisik dari RTC dapat dilihat pada Gambar 2.13.

Pada penelitian ini menggunakan RTC DS3231 yang memiliki spesifikasi seperti pada Tabel 2.8 berikut :

Tabel 2.8 Spesifikasi RTC DS3231

Power Input	5V
Jumlah Pin	8 buah
Low power operation	2.0V~5.5V
Temp Range	-40°C - 85°C
Memory (RAM)	31 x 8 byte
Max Time	2100 (Year)

Sumber : (www.Maximintergrated.com)



Gambar 2.13 RTC DS3231

Sumber : (http://www.dx.com)

BAB 3 METODOLOGI

Penelitian akan diwali dengan studi literatur serta kajian pustaka. Penelitian ini bersifat implementatif yaitu dengan melakukan penerapan metode *wake up interrupt* untuk *low power mode* pada sensor *node*. Langkah-langkah penelitian dapat diGambarkan pada Gambar 3.1

Gambar 3.1 Langkah-Langkah Penelitian

3.1 Studi Literatur

Studi literatur yaitu mempelajari mengenai penjelasan dasar teori yang digunakan untuk menunjang penelitian ini. Dasar-dasar teori tersebut diperoleh dari buku, jurnal, e-*book*, dan dokumentasi project. Dasar-dasar teori yang digunakan pada penelitian ini, diantaranya:

1. Cara kerja Protokol TDMA dan TPSN
2. Low Power Mode
3. Mikrokontroler Arduino Pro Mini
4. Chip Wireless nRF24l01
5. Sensor Suhu dan Kelembapan DHT11
6. Real Time Clock

Tujuan dari studi literatur adalah untuk mendapatkan dasar teori sebagai acuan untuk penulisan skripsi dan pengembangan aplikasi.

3.2 Analisis Kebutuhan

Analisis kebutuhan sistem bertujuan untuk memperoleh semua kebutuhan yang diperlukan untuk membangun sistem yang diuji pada penelitian ini. Analisis kebutuhan pada penelitian ini dibagi menjadi sebagai berikut.

3.2.1 Kebutuhan Fungsional

Jenis kebutuhan yang berisi proses-proses apa saja yang nantinya dilakukan oleh sistem. Kebutuhan fungsional juga berisi informasi-informasi apa saja yang harus ada dan dihasilkan oleh sistem. Kebutuhan fungsional dari sistem ini diantaranya:

1. Masing-masing *wireless sensor node* akan memiliki waktu yang tersinkronisasi antara satu dengan yang lain dengan metode TPSN.
2. Waktu penjadwalan lebih disesuaikan bedasarkan hasil sinkronisasi TPSN.
3. Masing-masing *wireless sensor node* dapat mengirimkan data sesuai dengan metode penjadwalan yang sudah dibuat berdasarkan protokol TDMA.
4. *Wireless sensor node* dapat menggunakan daya yang lebih kecil ketika sedang dalam keadaan *sleep* sehingga mengurangi konsumsi daya berlebih.

3.2.2 Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras (*hardware*) adalah menganalisis semua kebutuhan perangkat keras yang akan digunakan untuk membangun sistem yang akan diteliti pada penelitian ini. Perangkat keras yang digunakan antara lain :

1. Laptop
2. Arduino Pro Mini
3. Modul *Wireless* NRF24L01
4. Sensor suhu DHT11
5. Sensor LDR
6. Sensor Rain Drop
7. Sensor Soil Moisture
8. Real Time Clock DS1302
9. FTDI232 Serial Bus
10. Kabel USB
11. Baterai 9 Volt

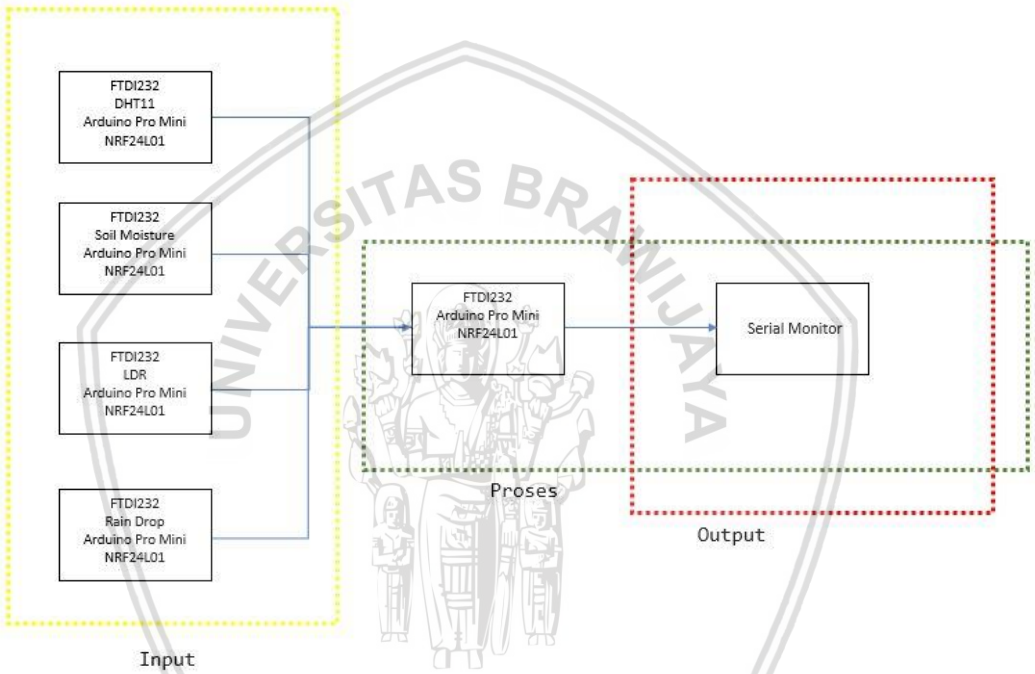
3.2.3 Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak adalah menganalisa semua kebutuhan dari sisi perangkat lunak yang akan digunakan untuk membuat kode program yang akan diterapkan pada sistem. Perangkat lunak yang digunakan antara lain :

1. Arduino IDE 1.8.2
2. *Library* Arduino

3.3 Perancangan Sistem

Tahap perancangan sistem merupakan langkah awal untuk membuat sebuah sistem. Pada tahap perancangan dibuat dalam sebuah diagram blok yang dimana nantinya berisi bagian-bagian dari sistem yang dibuat. Perancangan dari sistem yang dibuat secara keseluruhan dapat dilihat pada Gambar 3.2



Gambar 3.2 Block Diagram Sistem

Berdasarkan Gambar Gambar 3.2 perancangan sistem penelitian dapat dijelaskan sebagai berikut :

1. Fase *input*
   Fase ini berjalan pada *node transmitter*. Dimana *node transmitter* melakukan sinkronisasi waktu dengan *node receiver* serta penetapan jadwal pengiriman data menuju *node receiver.* Setelah itu barulah *node* akan berjalan sesuai dengan jadwal. *Sensor* yang ada pada *node transmitter* akan mengakuisisi data lalu melalui media *transceiver* data tersebut akan dikirim menuju *node receiver* jika sudah memasuki jadwal untuk mengirimkan data. Ketika sudah melewati jadwal pengiriman data yang sudah ditentukan, selanjutnya *node transmitter* akan memasuki mode *sleep* untuk menghemat daya hingga tiba jadwalnya untuk kembali mengirimkan data.
2. Fase Proses

Fase proses berjalan pada *node receiver,*dimana data yang di terima berasal dari *node-node transmitter* yang ada pada jaringan sesuai dengan jadwal yang telah ditentukan. Pada fase proses *serial monitor* akan menampilkan data yang telah diproses.

3. Fase *output*
   Fase *output* berjalan pada *node receiver.* Dimana *node receiver* akan menampilkan data yang sudah diterima dan diolah sedemikian rupa untuk ditampilkan pada *serial monitor* dari *node receiver*.

3.4 Implementasi

Implementasi merupakan proses penyusunan perangkat lunak bedasarkan kebutuhan yang telah ditentukan. Pada tahap ini diharapkan sistem yang telah dirancang bisa dijalankan pada keadaan yang sebenarnya, sehingga akan diketahui apakah sistem yang dibuat benar-benar dapat digunakan dan sesuai dengan tujuan yang diinginkan.

Langkah-langkah untuk melakukan implementasi antara lain:

1. Merancang dan merakit rangkaian sensor pada *wireless sensor node*.
2. Mengirimkan data ke *base station* untuk memastikan rangkaian sudah sesuai dengan perancangan.
3. Membuat algoritma penjadwalan pengirman dengan Metode TDMA.
4. Menerapkan mode Low Power pada *wireless sensor node*.
5. Menampilkan data yang diterima *base station* pada serial monitor.

3.5 Pengujian Sistem

Pengujian akan dilakukan dalam beberapa tahapan yaitu pengujian fungsional dan pengujian akurasi. Pengujian fungsional berfungsi untuk melihat apakah kebutuhan fungsional sistem berjalan sesuai dengan perancangan atau tidak. Sedangkan pengujian akurasi berfungsi untuk melihat bagaimana data yang ditampilkan ke pengguna oleh sistem telah akurat untuk selanjutnya dilakukan analisis terhadap data tersebut. Pengujian sistem dijelaskan pada poin-poin berikut ini :

Rancangan skenario pengujian sistem akan dideskripsikan pada poin-poin berikut ini :

1. Pengujian pada sisi *wireless sensor node* (*Transmitter*)
   a. *sensor* dapat mengakuisisi data nilai suhu,cahaya, kelembaban tanah serta hujan.
   b. Dapat melakukan sinkronisasi waktu antar *node-node* yang lain sesuai dengan metode *Timing-Sync Protocol for Sensor Network*.
   c. Dapat melakukan pengiriman sesuai dengan slot waktu yang telah ditetapkan menggunakan metode *Time Division Multiple Access*. (TDMA).

d. *Node-node* dapat memasuki *low power mode* sesuai dengan jadwal yang telah ditentukan dengan perhitungan waktu berdasarkan jam RTC dengan metode *wake up interrupt*.
e. Perbandingan perbedaan arus pada tiap *node* saat melakukan pengamatan serta pada saat memasuki *low power mode*.

2. Pengujian pada sisi *base station* (*Receiver*)
   a. Dapat menerima data dari tiap *wireless sensor node* sesuai dengan parameter alokasi waktu yang ditentukan.
   b. Data dapat diolah dan diproses sesuai dengan parameter yang telah ditententukan.
   c. Data yang diterima dapat ditampilkan melalui *serial monitor*.

3.6 Analisis Hasil

Setelah data diperoleh dari hasil pengujian, akan dilakukan analisis secara kuantitatif untuk mengukur kinerja sistem apakah sistem sudah berjalan dengan baik dan sesuai dengan tujuan penelitian. Kesimpulan dapat ditarik dari analisis terhadap pengujian yang dilakukan.

BAB 4 REKAYASA KEBUTUHAN

Pada bab ini berisi penjelasan tentang persyaratan-persyaratan yang harus dipenuhi agar sistem yang dibuat dapat berjalan secara benar dan berfungsi sesuai dengan tujuan. Rekayasa kebutuhan akan dijelaskan secara rinci untuk mencapai hasil penelitian yang diharapkan terkait dengan implementasi *low power* pada wireless sensor *node* untuk akuisisi data lahan pertanian dengan metode *wake up interrupt* modul RTC.

4.1 Deskripsi Umum

4.1.1 Perspektif Sistem

Sistem ini akan bekerja sesuai tujuan jika setiap *node sensor* dapat melakukan *wake up interrupt* untuk melakukan pengiriman data-data sensor dengan arus rendah sesuai dengan penjadwalan.

4.1.2 Tujuan

Tujuan dari bab ini adalah menjelaskan sistem penelitian secara rinci dan detail mulai dari komponen yang dibutuhkan sistem dan bagaimana sistem akan bekerja serta beroperasi secara baik. Sehingga didapatkan dokumentasi yang cukup jelas mengenai alur pembuatan sistem dan tujuan dari sistem. Dokumentasi ini ditujukkan untuk pengembangan sistem selanjutnya serta diusulkan sebagai persetujuan laporan skripsi.

4.1.3 Ruang Lingkup

Dalam sistem wireless sensor *node* yang diajukan ini dirancang untuk bisa melakukan *wake up interrrupt* sesaat sebelum *sensor node* melakukan *sensing*. Setiap *node* diharapkan mampu bekerja dengan arus rendah sesuai dengan waktu yang dijadwalkan.

Pembuatan sistem ini ditujukan kepada penguji,pengembang dan pengguna yang membutuhkan sehingga memungkinkan untuk dikembangkannya sistem ini lebih lanjut untuk fungsi yang lebih kompleks serta untuk keperluan ditingkat yang lebih rumit.

4.1.4 Batasan Sistem

Sistem ini memiliki batasan-batasan antara lain :

1. Arsitektur perangkat keras dibuat berdasarkan standar arsitektur wireless sensor *node*.
2. Waktu yang digunakan adalah waktu berdasarkan perhitungan dari modul *Real Time Clock.*
3. Menggunakan 1 buah *node* sebagai *receiver* dan 4 buah *node* sebagai *transmitter*.

4. Pengurangan arus hanya dilakukan pada tiap *node transmitter* saat keadaan *sleep/idle.*
5. *Interrupt* oleh modul RTC DS3231 dilakukan sesuai dengan waktu yang sudah ditentukan pada kode program.
6. *Interrupt* berfungsi sebagai alarm untuk membangunkan *node* yang sedang dalam keadaan *sleep*.
7. Rancangan perangkat keras menggunakan rancangan pada penelitian sebelumnya, sehingga performanya dapat diasumsikan sama dengan yang sebelumnya.

4.1.5 Asumsi dan ketergantungan

Beberapa asumsi dan ketergantungan dalam sistem ini antara lain :

1. Besar arus listrik akan bergantung pada arsitektur dan desain dari sistem yang dibuat pada tiap *wireless sensor node*.
2. Besar arus juga akan dipengaruhi oleh program yang digunakan pada sistem.
3. Waktu akan tersinkronisasi antara *receiver* dan *transmitter* diasumsikan akan sinkron.

4.2 Rekayasa Kebutuhan

Pada bagian subbab ini akan dijelaskan tentang rekayasa kebutuhan sistem yang meliputi kebutuhan fungsional , kebutuhan eksternal serta kebutuhan lainnya yang menunjang sistem agar berjalan sesuai dengan kebutuhan.

4.2.1 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan yang harus terpenuhi sehingga sistem dapat berjalan sesuai dengan tujuan. Berikut merupakan kebutuhan fungsional dari sistem yang dibuat, antara lain :

1. Fungsi pembacaan sensor-sensor

   Fungsi ini mengharuskan tiap-tiap sensor dapat bekerja dengan baik dan berhasil melakukan *sensing* terhadap lingkungan penelitian.

2. Fungsi *wireless* pada *transmitter* dan *receiver*

   Fungsi ini mengharuskan pengiriman serta penerimaan data antara *transmitter* dan *receiver* berjalan dengan baik sesuai dengan sinkronisasi waktunya.

3. Fungsi *low power mode* pada *transmitter*

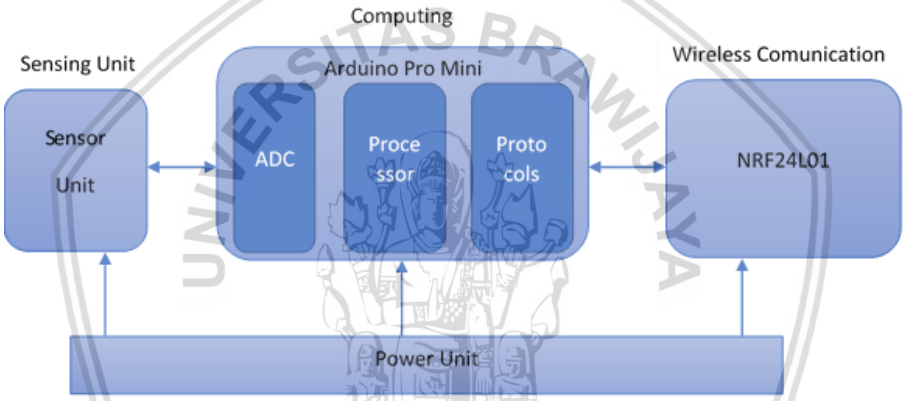
   Fungsi ini merupakan fungsi yang melakukan penghematan penggunaan daya pada *mikrokontroller* saat tidak sedang melakukan fungsinya.

4. Fungsi *interrupt* pada *transmitter*

   Pada fungsi ini *interrupt* dari modul RTC sudah harus dapat bekerja sesuai dengan waktu yang telah ditentukan pada kode program. Selain itu setiap *node* harus dapat bekerja dengan tingkat energi yang rendah agar dapat menyesuaikan penggunaan energi sesuai jadwal. Fungsi ini merupakan tujuan utama dari penelitian ini.

4.2.2 Kebutuhan Perangkat Keras

Dalam pembuatan penelitian ini dibutuhkan beberapa perangkat keras yang dirancang untuk dapat berjalan sesuai tujuan. Pada *transmitter* dan *receiver* masing-masing memiliki perangkat keras spesifik untuk menujang jalannya sistem. Pada Gambar 4.1 akan menunjukan secara menyeluruh perangkat keras dari *transmitter* maupun *receiver*.



Gambar 4.1 Kebutuhan Perangkat Keras

1. *Node Transmitter* :
   a. Mikrokontroler Arduino Pro Mini berfungsi sebagai modul pemrosesan data utama dari data hasil *sensing* dari tiap-tiap sensor. Data-data itu akan diolah dan dikirim ke *receiver* melalui komunikasi *wireless*.
   b. Sensor- sensor pertanian terdiri atas DHT11, LDR, *Rain Drop*, dan *Soil Moisture* digunakan untuk mendapatkan data keadaan lingkungan pertanian.
   c. NRF24L01 berfungsi sebagai modul komunikasi wireless antar *receiver* dan *transmitter*.
   d. *Real Time Clock* berperan sebagai pengatur waktu pada tiap *node*, serta memberikan interrupt pada waktu tertentu.
2. *Node Receiver* :
   a. Mikrokontroler Arduino Pro Mini pada receiver berfungsi untuk mengolah data yang diterima dari transmitter untuk kemudian ditampilkan pada serial monitor.

b. NRF24L01 berfungsi sebagai modul komunikasi wireless antar *receiver* dan *transmitter*.

c. FTDI berfungsi sebagai modul komunikasi serial antara *node* dengan komputer untuk menampilkannya pada serial monitor.

d. Display serial monitor berfungsi untuk menampilkan data yang sudah diterima.

4.2.3 Kebutuhan Perangkat Lunak

Dalam penelitian ini dibutuhkan perangkat lunak yang terdiri atas library-library yang digunakan dalam Arduino IDE yang memiliki fungsi khusus yang menunjang jalannya sistem. Pada Gambar 4.2 menjelaskan tentang kebutuhan perangkat lunak pada *transmitter* maupun *receiver*.

1. *Node Transmitter* :

   a. Arduino IDE berfungsi untuk menulis serta upload kode program ke mikrokontroler Aduino Pro Mini.

   b. MIRF Library adalah library yang berfungsi untuk melakukan komunikasi secara *wireless* melalui NRF24L01 antar *node*.

   c. Jeelib Library merupakan library *custom* pada arduino yang membuat mikrokontroler dapat menjalankan *low power mode*.

   d. TPSN merupakan algoritma yang berfungsi untuk mensiknronisasi waktu antar *node transmitter* maupun *receiver*.

   e. TDMA merupakan algoritma yang berfungsi mengatur penjadwalan pengiriman data dari *node transmitter* ke *node receiver*.

   f. RTC library berfungsi sebagai pemberi *interrupt* pada tiap-tiap *node* berdasarkan waktu yang sebenarnya.

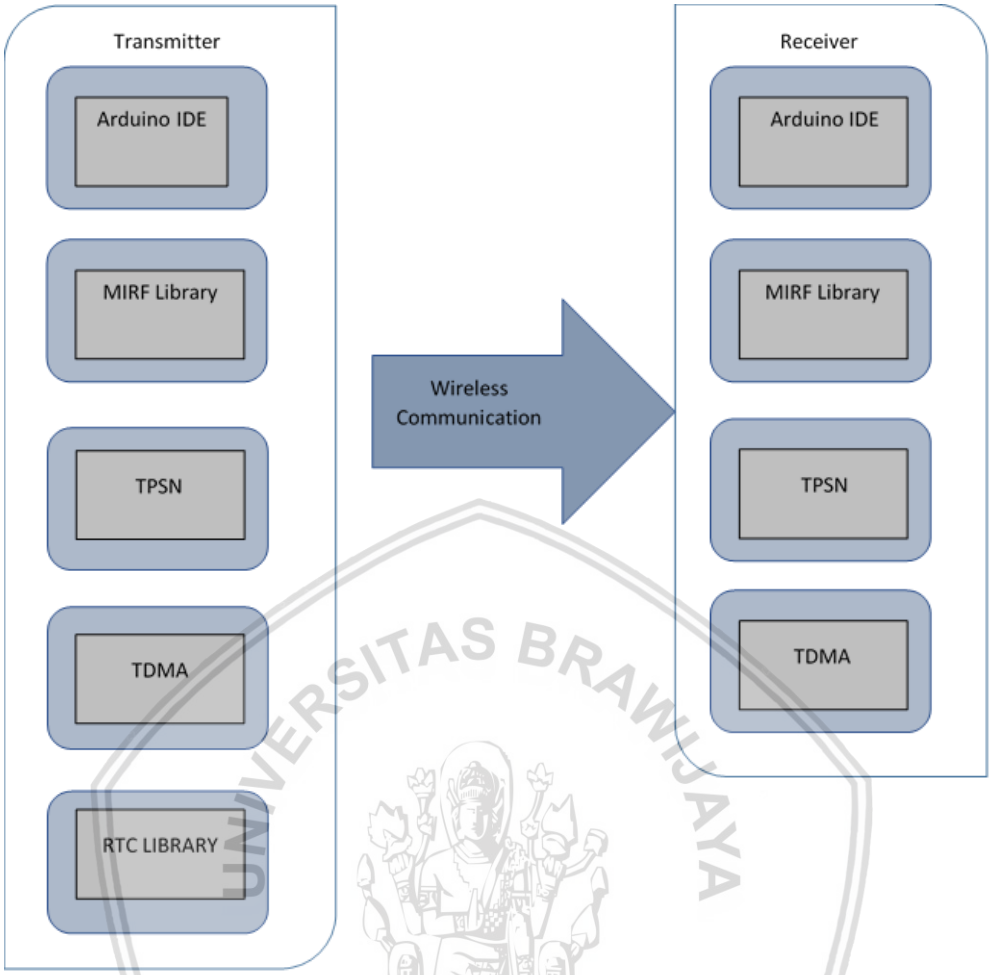
2. *Node Receiver* :

   a. Arduino IDE berfungsi untuk menulis serta upload kode program ke mikrokontroler Aduino Pro Mini.

   b. MIRF Library adalah library yang berfungsi untuk melakukan komunikasi secara *wireless* melalui NRF24L01 antar *node*.

   c. TPSN merupakan algoritma yang berfungsi untuk mensiknronisasi waktu antar *node transmitter* maupun *receiver*.

   d. TDMA merupakan algoritma yang berfungsi mengatur penjadwalan pengiriman data dari *node transmitter* ke *node receiver*.



Gambar 4.2 Kebutuhan Perangkat Lunak

4.2.4 Kebutuhan Komunikasi Wireless

Modul *wireless* yang digunakan dalam penelitian ini adalah NRF24L01 yang mengunakan gelombang radio 2.4 GHz. Modul ini digunakan pada *transmitter* maupun *receiver*. Untuk menangani komunikasinya, pada Arduino IDE digunakan library khusus yaitu MIRF *Library*. Komunikasi antara *transmitter* dan *receiver* dibagi ke dalam beberapa alamat dan *channel* yang terpisah.

4.2.5 Batasan Desain Sistem

1. Desain *hardware* dari sistem dibuat berdasarkan desain dari penelitian sebelumnya, sehingga dapat diasumsikan memiliki performansi yang sama.
2. Pada sensing unit ditambahkan beberapa sensor seperti LDR, *Rain Drop Sensor* dan *Soil Moisture Sensor* yang digunakan untung mengakuisisi data pada lahan pertanian.
3. *Processing unit* pada tiap *node* menggunakan Arduino Pro Mini yang mengatur seluruh kinerja *node*.

4. Untuk *serial communication* digunakan FTDI yang berfungsi untuk mengupload kode program dari laptop ke unit mikrokontroler.
5. Untuk media tranceiver digunakan NRF24L01 yang menggunakan gelombang radio 2.4 GHz.
6. Pada tiap *node transmitter* ditambahkan modul *Real Time Clock* yang berfungsi memberikan fungsi *wake up interrupt* pada saat *low power* mode.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan Sistem

Pada subbab ini akan dijelaskan tentang perancangan perangkat keras, perangkat lunak , algoritma penjadwalan dan algoritma *wake up interrupt*.

5.1.1 Perancangan Perangkat Keras

5.1.1.1 Perancangan Perangkat Keras *Node Receiver*

Node receiver bertugas untuk menerima data dari tiap-tiap *node transmitter. Node* ini terdiri atas Arduino Pro Mini , modul *wireless* NRF24L01 dan FTDI sebagai penghubung ke laptop yang berguna untuk melakukan *upload* kode program serta memonitor hasilnya pada *serial monitor*. Pin-pin yang digunakan ada pada Tabel 5.1.

Tabel 5.1 Pin Arduino Pro Mini

Pin Analog	A0-A3
Pin Digital	D2-D13
Arus per pin	40 mA

Pada *node receiver* digunakan modul NRF24L01 yang berfungsi sebagai modul penerima data secara *wireless* dari setiap *node* transmitter. Pada Tabel 5.2 akan menampilkan keterangan dari pin modul NRF24L01.

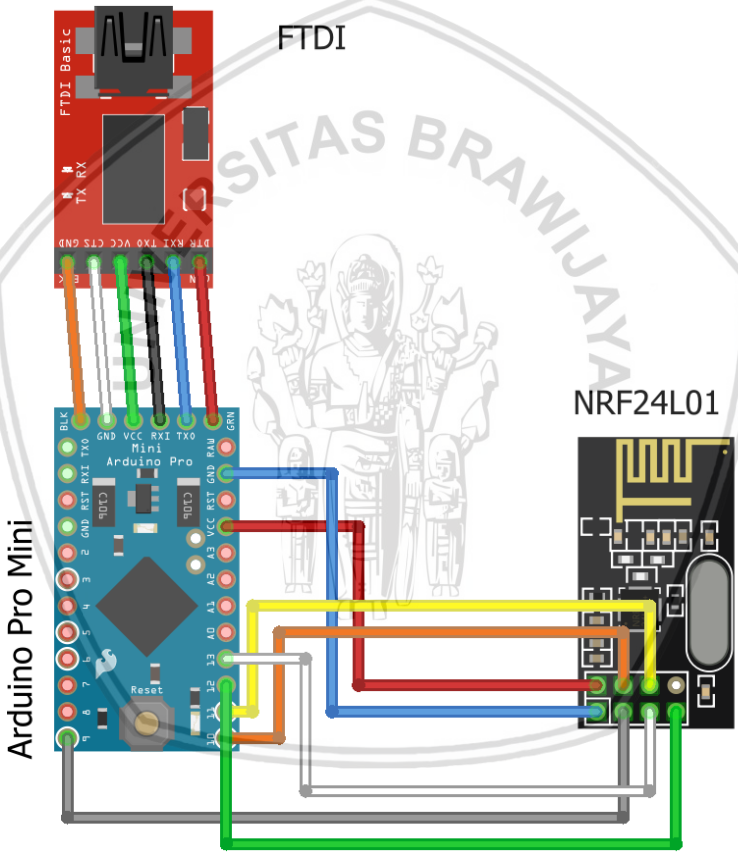
Tabel 5.2 Pin NRF24L01

VCC	Pin Power
GND	Pin Ground
CSN	SPI Chip Select
CE	Rx/Tx Enabler
MOSI	SPI data in
MISO	SPI data out
SCK	SPI Clock
IRQ	Pin Interrupt

Untuk menghubungkan mikrokontroller dengan komputer dibutuhkan *serial communicator* yaitu FTDI232. Modul ini berfungsi untuk alat komunikasi serial ke komputer agar hasil dari *node* dapat dilihat melalu *serial monitor* Arduino. Pada Tabel 5.3 dapat dilihat semua pin dari modul ini.

Tabel 5.3 Pin FTDI232

DTR	Pin Data Terminal Ready
Rx	Pin Receive
Tx	Pin Transmit
VCC	Pin Power
CTS	Pin Clear to Send
GND	Pin ground



Gambar 5.1 Rangkaian *Node Receiver*

Pada Gambar 5.1 terlihat modul NRF24L01, Arduino Pro Mini, dan FTDI232 harus dirangkai sedemikian rupa agar dapat berfungsi dengan tujuannya masing-masing. Pada NRF24L01 pin GND dan VCC yang merupakan sumber daya untuk modul ini akan dihubungkan pada pin GND dan VCC pada Arduino Pro Mini, pin CE dengan pin D9, pin CS dengan pin D10,pin MOSI dengan pin D11, pin SCK dengan pin D13, dan pin MISO dengan pin D12. Pada FTDI232 pin GND dihubungkan dengan pin BLK pada Arduino Pro Mini, pin CTS dengan pin GND, pin VCC dengan pin VCC, pin Tx dengan pin RX, pin Rx dengan pin Tx serta pin DTR dengan pin GRN.

5.1.1.2 Perancangan Perangkat Keras *Node Transmitter*

Node transmitter memiliki fungsi sebagai *node* yang melakukan pengambilan data terhadap lingkungan sekitar dan mengirimnya ke *node receiver. Node transmitter* terdiri atas sensor suhu dan kelembapan, sensor *rain drop*, sensor *soil moiture* dan sensor cahaya, semua sensor tersebut memiliki fungsi yang sesuai dengan namanya. Mikrokontroler, *serial communication* serta modul *wireless* yang digunakan pada *node* transmitter sama dengan yang ada pada *node* receiver yaitu Arduino Pro Mini , FTDI232, dan NRF24L01.Selain itu *node* transmitter dilengkapi oleh modul RTC DS3231 untuk melakukan penghitungan waktu dan *wake up interrupt*.

Selain sensor-sensor tersebut, *node* transmitter juga membutuhkan mikrokontroler untuk melakukan pemrosesan data. Mikrokontroler yang digunakan adalah Arduino Pro Mini dengan spesifikasi dari pin yang terdapat pada Tabel 5.4.

Tabel 5.4 Pin Arduino Pro Mini

Pin Analog	A0-A3
Pin Digital	D2-D13
Arus per pin	40 mA

Untuk melakukan pengiriman data ke *node receiver* , *node transmitter* juga menggunakan modul *wireless* NRF24L01 yang memiliki konfigurasi pin seperti pada Tabel 5.5.

Tabel 5.5 Pin NRF24L01

VCC	Pin Power
GND	Pin Ground
CSN	SPI Chip Select
CE	Rx/Tx Enabler
MOSI	SPI data in
MISO	SPI data out
SCK	SPI Clock
IRQ	Pin Interrupt

Untuk melakukan komunikasi dengan laptop serta untuk mengupload program ke mikrokontroler, *node transmitter* menggunakan FTDI232 yang memiliki susunan pin yang ditunjukkan oleh Tabel 5.6.

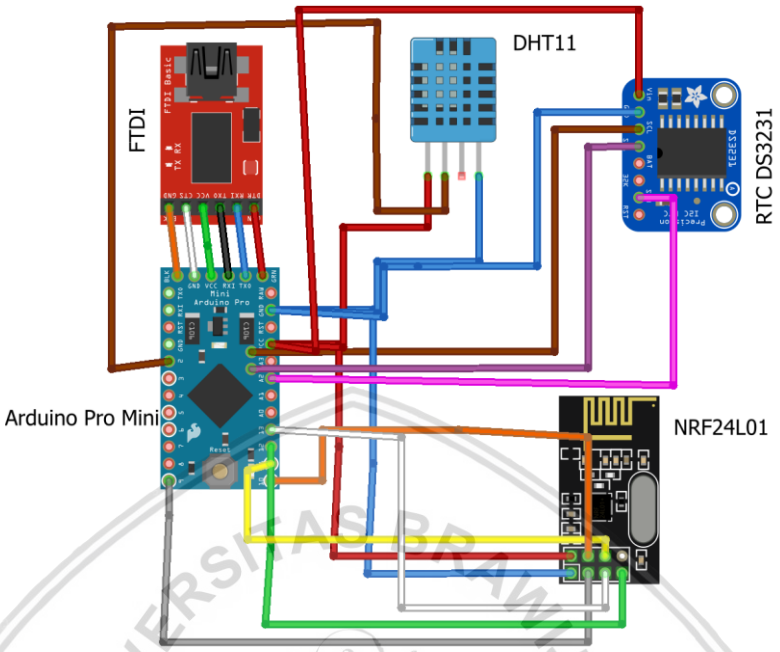
Tabel 5.6 Pin FTDI232

DTR	Pin Data Terminal Ready
Rx	Pin Receive
Tx	Pin Transmit
VCC	Pin Power
CTS	Pin Clear to Send
GND	Pin ground

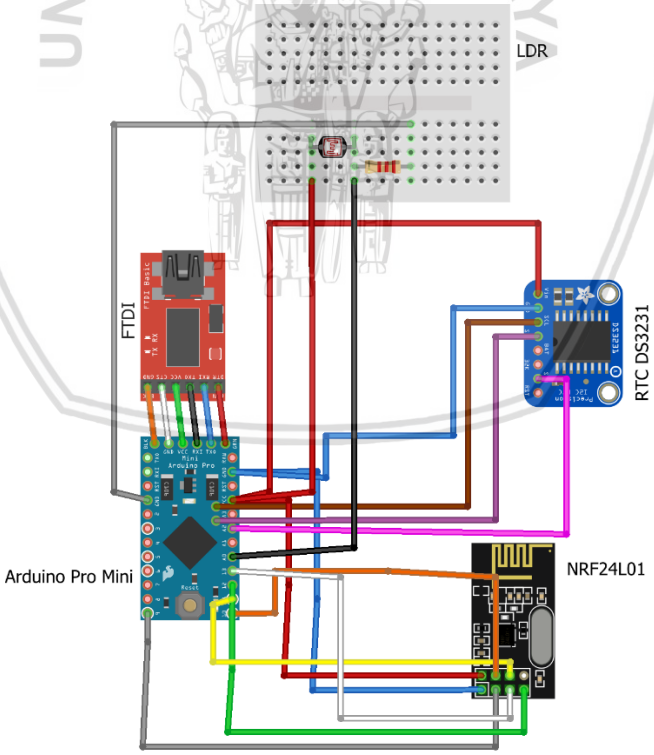
Selain itu, pada *node transmitter* terdapat modul RTC DS3231 yang memiliki fungsi utama sebagai pengitung waktu serta untuk melakukan *wake up interrupt* pada tiap-tiap *node*. Pada Tabel 5.7 merupakan konfigurasi pin dari RTC DS3231.

Tabel 5.7 Pin RTC DS3231

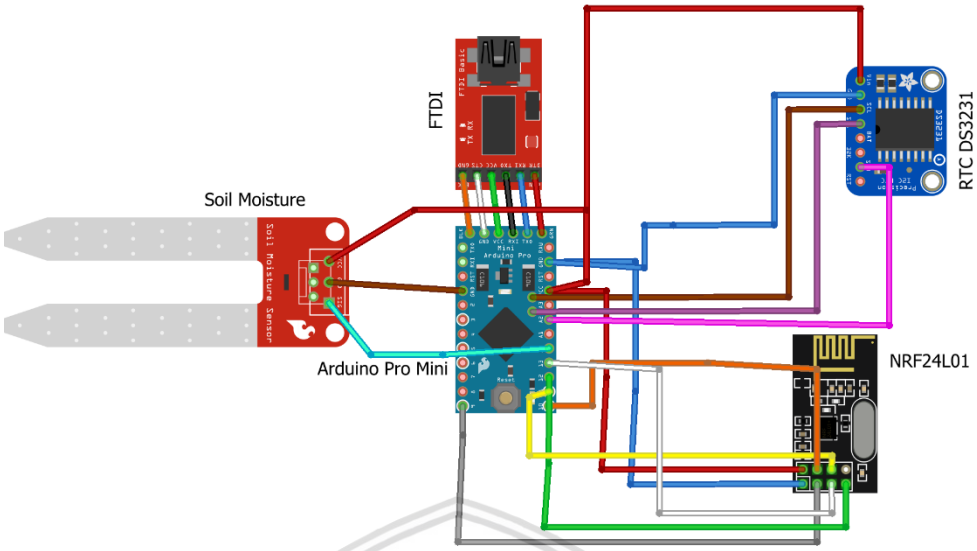
VCC	Pin Power
GND	Pin Ground
SDA	Pin Serial Data
SCL	Pin Serial Clock
SQW	Pin Square Wave output
32K	32K Oscillator output



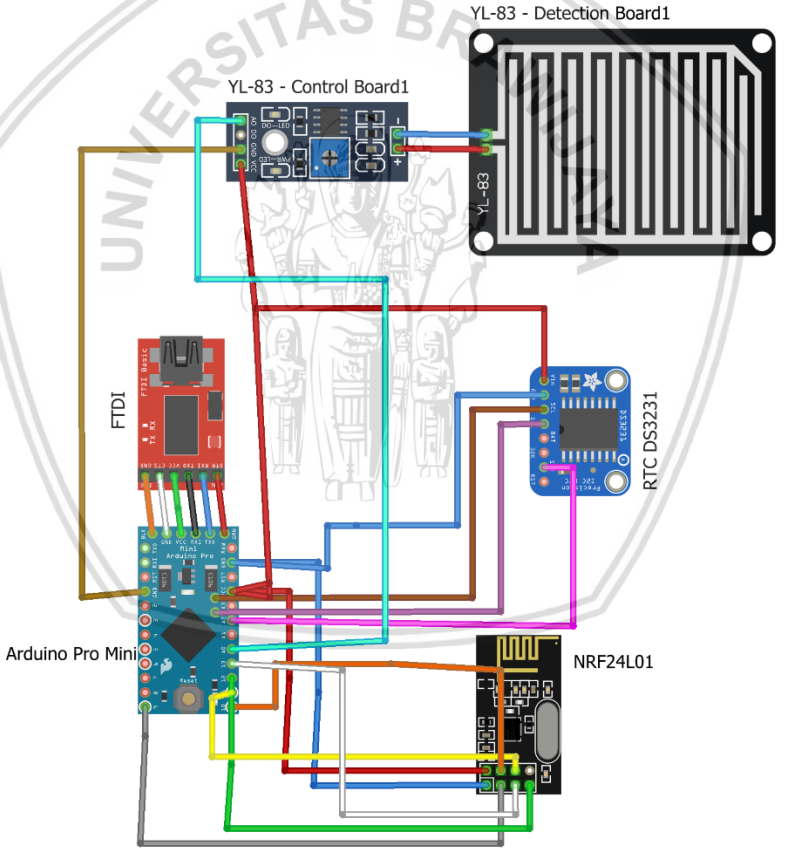
Gambar 5.2 Rangkaian *Node Transmitter* Dengan sensor DHT11



Gambar 5.3 Rangkaian *Node Transmitter* Dengan Sensor LDR



Gambar 5.4 Rangkaian *Node Transmitter* Dengan Sensor Soil Moisture



Gambar 5.5 Rangkaian *Node Transmitter* Dengan Sensor Rain Drop

Pada Gambar 5.2, 5.3 , 5.5 ,dan 5.5 dapat dilihat jika modul NRF24L01, Arduino Pro Mini, FTDI232, sensor-sensor serta RTC DS3231 harus dirangkai sedemikian rupa agar dapat berfungsi dengan tujuannya masing-masing. Pada NRF24L01 pin GND dan VCC yang merupakan sumber daya untuk modul ini akan dihubungkan pada pin GND dan VCC pada Arduino Pro Mini, pin CE dengan pin D9, pin CS dengan pin D10,pin MOSI dengan pin D11, pin SCK dengan pin D13, dan pin MISO

dengan pin D12. Pada FTDI232 pin GND dihubungkan dengan pin BLK pada Arduino Pro Mini, pin CTS dengan pin GND, pin VCC dengan pin VCC, pin Tx dengan pin RX, pin Rx dengan pin Tx serta pin DTR dengan pin GRN. Sedangkan pada RTC hanya 5 pin yang dipakai, yaitu pin VCC yang dihubungkan dengan pin VCC, pin GND dengan pin GND, pin SDA dengan pin A4, pin SCL dengan pin A5 dan pin SQW dengan pin A2.

5.1.1.3 Perancangan Simulasi *Wireless Sensor Network*

Simulasi untuk sistem akan dilakukan dengan menggunakan 5 buah *node* yang terdiri atas 1 *node receiver* yang dihubungkan ke laptop dan 4 *node transmitter* yang akan menggunakan baterai 9V sebagai sumber dayanya. Pada semua *node*, baik *receiver* maupun *transmitter* akan diberi program yang akan berjalan sesuai dengan fungsinya masing-masing. Kemudian setiap *node transmitter* akan disebarkan ditempat yang tidak terlalu jauh dari satu *node* dengan yang lain. Pada *node receiver* yang terhubung dengan komputer akan menampilkan data yang diterima dari *node transmitter* melalui *serial monitor.* Selain itu untuk pengukuran perbedaan arus saat *node transmitter* pada mode *low power/sleep* dibutuhkan multimeter.

5.1.2 Perancangan Perangkat Lunak

Pada subbab ini akan membahas mengenai perancangan perangkat lunak yang digunakan di dalam sistem. Perangkat lunak utama yang digunakan baik pada *node receiver* maupun *node transmitter* adalah Arduino IDE yang berfungsi untuk melakukan *compile* pada kode program yang digunakan oleh sistem. Selain penggunaan library yang ada di Arduino IDE dan library tambahan juga diperlukan agar sistem dapat berjalan dengan baik dan sesuai dengan tujuannya.

5.1.2.1 Perancangan Perangkat Lunak *Node Receiver*

Pemrograman yang digunakan untuk sistem ini menggunakan Arduino IDE sebagai software pengolah kode program yang berbasis bahasa C. Arduino IDE berfungsi untuk melakukan kompilasi serta menulis kode program untuk kemudian di upload ke *node receiver.*

Pada *node receiver* terdapat beberapa *library* yang berguna untuk menunjang jalannya program. Adapun *library* yang digunakan yaitu *library* SPI yang berguna untuk mengaktifkan komunikasi serial serta *library* MIRF yang berfungsi saat menggunakan modul *wireless* NRF24L01. Selain itu penggunaan library standar dari Arduino IDE juga digunakaan pada pemrograman *node receiver* ini.

5.1.2.2 Perancangan Perangkat Lunak *Node Transmitter*

Perancangan untuk perangkat lunak pada *node transmitter* memiliki kesamaan dalam hal bahasa pemrograman maupun pengunaan Arduino IDE seperti pada *node receiver.* Namun perbedaan mendasarnya adalah penggunaan *library* yang berbeda sesuai dengan penggunaan perangkat keras yang berbeda pula pada tiap *node* tersebut.

Library yang digunakan pada *node transmitt*er adalah library SPI yang berfungsi untuk mengaktifkan komunikasi serial, *library* MIRF yang berfungsi untuk memanfaatkan komunikasi melalui modul wireless NRF24L01. Selain itu juga ada beberapa penggunaan *library* untuk sensor DHT11, LDR, *Soil Moisture*, *Raindrop* dan RTC DS3231.

5.1.3 Perancangan Algoritme

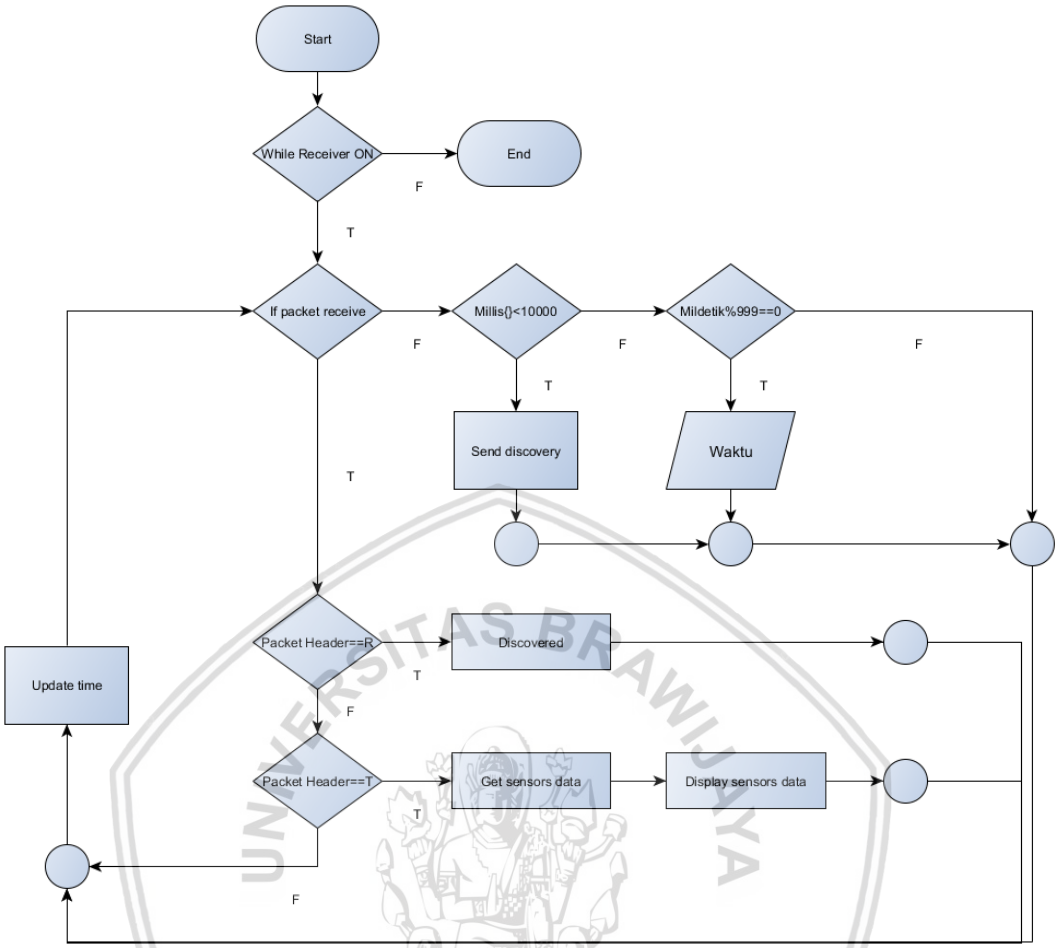
Pada subbab ini akan membahas diagram alir untuk algoritma yang digunakan pada *node receiver* dan *node transmitter*.

5.1.3.1 Perancangan Algoritme *Node Receiver*

Node Receiver pada sistem berguna sebagai penerima data dari *node transmitter* serta menampilkannya pada *serial monitor.* Penjelasan berupa flowchart dapat dilihat pada Gambar 5.3.

Sebelum melakukan komunkasi dengan *node-node* lainnya, *node receiver* akan melakukan inisialisasi *baudrate*, pin, *payload*, *channel* dan waktu tunggu. Pada inisialisasi *baudrate* mikrokokontroler akan melakukan penyesuaian *rate* kerjanya. Inisialisasi pin akan menyesuaikan pin yang digunakan pada tiap-tiap modul yang terpasang. Inisialisasi *channel* dilakukan agar tiap *node* dapat berkomunikasi satu dengan yang lain secara *wireless*. Kemudian pada inisialisasi *payload* akan ditentukan seberapa besar *payload* yang berisi protokol TPSN dan TDMA.

Setelah melakukan inisialisasi, *node receiver* akan melakukan pengiriman paket discovery ke tiap *node* transmitter. Jika semua *node* dapat menerima paket tersebut, maka tiap *node* transmitter dapat melakukan request sinkronisasi waktu. Paket discovery yang dikirim tersebut berisi alokasi waktu untuk *node transmitter* untuk jadwal pengiriman data sensor. Kemudian *receiver* akan mengirimkan paket konfirmasi sinkronisasi yang berisi konfirmasi dan waktu sinkronisasi agar *node transmitter* tersebut memiliki waktu yang sama dengan waktu yang dimiliki oleh *receiver*. Setelah waktu telah tersinkronisasi maka *node receiver* dapat mulai menerima data yang dikirimkan oleh *node transmitter* secara bergantian ketika waktu dan jadwal yang sudah ditentukan.



Gambar 5.6 *Flowchart Node Receiver*

5.1.3.2 Perancangan Algoritme *Node Transmitter*

Node Transmitter pada sistem berguna sebagai *node* yang melakukan akuisisi data dari tiap sensor. Penjelasan berupa flowchart dapat dilihat pada Gambar 5.4.

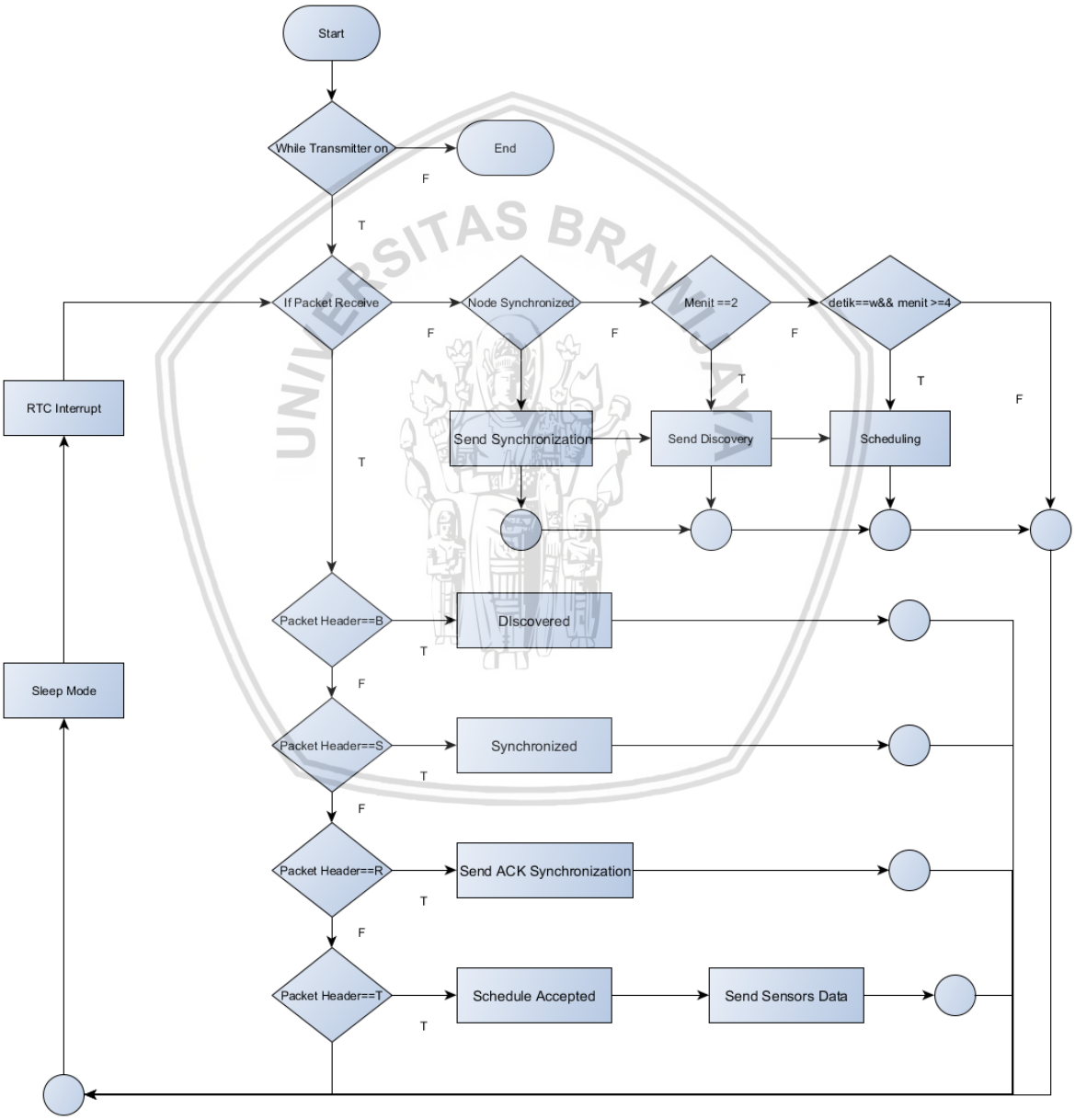
Sebelum melakukan komunikasi dengan *node-node* lainnya, *node transmitter* akan melakukan inisialisasi *baudrate*, pin sensor, *channel* dan waktu tunggu. Pada inisialisasi *baudrate* mikrokokontroler akan melakukan penyesuaian *rate* kerjanya. Inisialisasi pin akan menyesuaikan pin sensor yang digunakan pada tiap-tiap modul yang terpasang. Inisialisasi *channel* dilakukan agar tiap *node* dapat berkomunikasi satu dengan yang lain secara *wireless*

Setelah melakukan inisialisasi, *node* transmitter akan melakukan sinkronisasi waktu dengan protokol TPSN untuk memastikan jika tiap *node* menerima paket data atau tidak. Jika *node* menerima data, maka data tersebut merupakan paket *discovery* yang diberikan oleh *node receiver*. Paket tersebut juga berisi jadwal slot alokasi waktu yang akan menjadi jadwal untuk tiap *node transmitter*.

Saat seluruh *node* telah menerima pesan *discovery* kemudian akan memasuki fase *synchronizasation* dimana *node transmitter* akan menerima paket sinkronisasi waktu dari *node receiver.* Setelah menerima waktu yang ada pada

semua *node* akan menjadi menjadi setara. Kemudian *node* akan mengirim paket discovery sesuai dengan levelnya masing-masing, Level 1 akan memulai pengiriman paket untuk *node* pada Level 2 dan seterusnya.

Pengiriman data akan dimulai ketika sampai pada waktu yang telah ditentukan sebelumnya. Masing-masing *node* akan mengirimkan data ke *receiver* sesuai dengan alokasi waktu yang sudah tersimpan. Saat *node* telah selesai mengirimkan paket dengan jumlah tertentu , maka *node* akan menjalankan fungsi *interrupt* dari modul RTC dengan waktu yang ditentukan di dalam kode program, setelah itu tiap *node* akan memasuki *state sleep/low power.*



Gambar 5.7 Flowchart *Node* Transmitter

5.2 Implementasi Sistem

Subbab ini akan menjelaskan mengenai proses implementasi sistem dimulai dari perancangan perangkat keras dan perangkat lunak yang digunakan pada sistem.

5.2.1 Implementasi Perangkat Keras

5.2.1.1 Implementasi *Node Receiver*

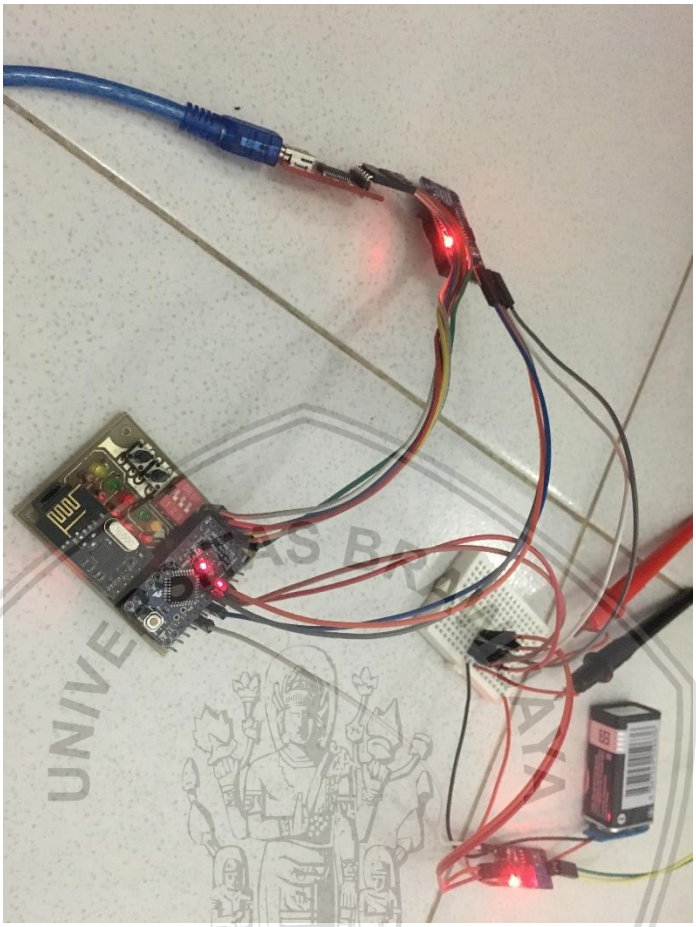
Sesuai pada perancangan pada subbab 5.1.1.1 , *node receiver* terdiri atas Arduino Pro Mini sebagai mikrokontroler dan NRF24L01 sebagai bagian komunikasi *wireless.* Hasil dari implementasi *node receiver* dapat dilihat pada Gambar 5.5.

Gambar 5.8 Implementasi Perangkat Keras *Node Receiver*

5.2.1.2 Implementasi *Node Transmitter*

Sesuai pada perancangan pada subbab 5.1.1.2 , *node transmitter* terdiri atas Arduino Pro Mini sebagai mikrokontroler ,NRF24L01 sebagai bagian komunikasi *wireless,* RTC sebagai pemberi metode *wake up interrupt* dan sensor-sensor yang terdiri atas DHT11, LDR, Rain Drop, dan Soil moisture yang spesifikasinya dapat

dilihat pada subbab tersebut . Hasil dari implementasi *node* transmitter dapat dilihat pada Gambar 5.6.



Gambar 5.9 Implementasi Perangkat Keras *Node Transmitter*

5.2.2 Implementasi Perangkat Lunak

5.2.2.1 Implementasi Perangkat Lunak *Node* Receiver

Node receiver dalam metode penjadwalan TDMA dan TPSN diimplementasikan sebagai *root* yang berada di level 0. Sama seperti *node transmitter*, *node receiver* menerapkan 2 prosedur untuk menyetarakan waktu. Pada tahap prosedur *discovery*, *root* akan mengirimkan paket *discovery* selama 10 detik secara *Broadcast*. Namun berbeda dengan *node transmitter*, *node receiver* tidak menerima paket *discovery* dari *node* lain karena posisinya yang berada di level 0 pada topologi. Kode program *discovery* pada *node receiver* dapat dilihat pada Tabel 5.17 berikut.

Tabel 5.8 Potongan Kode Sumber Method *discovery() Node Receiver*

Kode sumber Method *discovery()*	
1	`void discovery( ){`
2	`Mirf.setTADDR((byte *)"node");`
3	`strcpy(DataKirim,"B");`
4	`strcat(DataKirim,AlamatSendiri);`
5	`strcat(DataKirim,Level);`
6	`DataKirim[30]=q;`
7	`Mirf.send((byte *)&DataKirim);`
8	`while(Mirf.isSending()){`
9	`Serial.println("discovery message spreading across all node!");`
10	`delay(100);`
11	`}`
12	`}`

Kemudian *node receiver* akan mengirimkan paket yang berisi alamat *node* dan level *node* itu sendiri dengan paket header "B" ke semua *node* aktif yang belum menjadi *root*. Selain itu dikirim juga slot waktu yang akan diterima oleh *node transmitter* untuk waktu jadwal pengiriman data. Setelah mengirimkan paket *discovery*, *root* akan menunggu sampai ada permintaan sinkronisasi waktu dari *node* yang menerima paket *discovery*. ketika *receiver* telah menerima paket permintaan penyetaraan waktu, kemudian *receiver* akan membalas *node* tersebut dengan mengirimkan paket sinkronisasi waktu yang berisikan waktu lokal *receiver* dan jadwal pengiriman data untuk *node transmitter* tersebut. Implementasi kode program method *synchronize*() dapat dilihat pada Tabel 5.18.

Tabel 5.9 Potongan Kode Sumber Method synchronize() *Node Receiver*

Kode Sumber Method *synchronize*()	
1	`void synchronize(){`
2	`TDMAsched();`
3	`for (int x=0; x<4; x++){`
4	`AlamatTerima[x] = DataTerima[x+1];`
5	`}`
6	`for (int x=0; x<3; x++){`
7	`TimeStamp1[x] = DataTerima[x+5];`
8	`}`
9	`Serial.print("request synchronization from: ");`
10	`Serial.println(AlamatTerima);`
11	`cetakwaktu();`
12	`Mirf.setTADDR((byte *)AlamatTerima);//kirim balasan`
	`kirimwaktu();//timestamp3`

```
13      strcat(DataKirim,TimeStamp1);//timestamp1
14      if(TimeStampInt<10){
15       strcat(DataKirim,"0");
16       strcat(DataKirim,"0");
17       }else if(TimeStampInt<100){
18       strcat(DataKirim,"0");
19       }
20       strcat(DataKirim,TimeStamp2);//timestamp2
21       sprintf(Sched2,"%d", Sched);
22       strcat(DataKirim,Sched2);
23       Mirf.send((byte *)&DataKirim);
24       akb = millis();
25       akb = akb-akk;
26       Serial.println(akb);
27       while(Mirf.isSending()){
28       Serial.println("synchronization sent!");
29        }
  }
```

Setelah paket sinkronisasi terkirim, maka waktu *node receiver* dan *transmitter* akan setara. Pada saat itu barulah *node transmitter* dapat mengirimkan data *sensor* sesuai jadwal yang telah diberikan. *Receiver* akan menunggu sampai mendapatkan data dari *node transmitter*. Setelah mendapatkan paket yang berisikan data sensor, selanjutnya *node receiver* akan menampilkan data tersebut di *serial monitor*. Implementasi kode program untuk menerima dan menampilkan data *sensor* pada *node receiver* dapat dilihat pada method *TDMAack*() Tabel 5.19.

Tabel 5.10 Potongan Kode Sumber Method TDMAack() *Node Receiver*

Kode Sumber TDMAack()	
1	`void TDMAack(){`
2	`Mirf.getData((byte*)&DataTerima);`
3	`float suh = DataTerima[31];`
4	`Serial.print("temperature");`
5	`Serial.print(" : ");`
6	`Serial.print(suh);`
7	`Serial.println("oC");`
8	`}`

5.2.2.2 Implementasi Perangkat Lunak *Node Transmitter*

Node transmitter berfungsi sebagai *node* yang mengirimkan data dari tiap sensor dengan komunikasi secara *wireless* dan mampu melakukan sinkronisasi

waktu dengan *node receiver*. Oleh karena itu mikrokontroller Arduino pada *node transmitter* harus di program sedemikian rupa dan membutuhkan lebih dari *library* standar yang dimiliki oleh Arduino IDE. Beberapa *library* khusus digunakan pada program yang akan dijalankan oleh Arduino pada *node transmitter* untuk menunjang perangkat dan algoritma yang digunakan. Implementasi kode program *library* yang digunakan pada program *node transmitter* dapat dilihat pada Tabel 5.20.

Tabel 5.11 Potongan Kode Sumber Library *Node Transmitter*

Kode Sumber Library	
1	`#include <Ports.h>`
2	`#include <LowPower.h>`
3	`#include <DHT.h>`
4	`#include <SPI.h>`
5	`#include <Mirf.h>`
6	`#include <nRF24L01.h>`
7	`#include <MirfHardwareSpiDriver.h>`
8	`#include <stdio.h>`
9	`#include <string.h>`
10	`#include <stdlib.h>`
11	`#include <Wire.h>`
12	`#include <RTClibExtended.h>`

Node transmitter memiliki fungsi *sleep* yang dijalankan oleh modul RTC dengan memberikan metode *wake up interrupt* .Pada fungsi `RTC.setAlarm(ALM1_MATCH_HOURS)` merupakan salah satu cara pemanfaatan fitur alarm yang dimiliki oleh RTC DS3231 yaitu dengan menggunakan parameter jam dan menit. Jadi, saat waktu pada RTC sudah sesuai dengan waktu yang telah ditentukan pada kode program, maka RTC akan menjalankan fungsi *interruptnya* dan mikrokontroler akan memasuki *state sleep*. Pada Tabel 5.21 menunjukkan fungsi interrupt RTC yang digunakan pada program *node transmitter*.

Tabel 5.12 Potongan Kode Sumber Fungsi *Interupt* RTC *Node Transmitter*

Kode Sumber Interrutp RTC	
1	`void setup() {`
2	`  Serial.begin(9600);`
3	`  pinMode(wakePin, INPUT);`
4	`  Wire.begin();`
5	`  RTC.begin();`
6	`  RTC.adjust(DateTime(__DATE__, __TIME__));`
7	`  RTC.armAlarm(1, false);`

```
8     RTC.clearAlarm(1);
9     RTC.alarmInterrupt(1, false);
10    RTC.armAlarm(2, false);
11    RTC.clearAlarm(2);
12    RTC.alarmInterrupt(2, false);
13    RTC.writeSqwPinMode(DS3231_OFF);
14    RTC.setAlarm(ALM1_MATCH_HOURS, 21, 18, 0);
15    RTC.alarmInterrupt(1, true);
    }
```

Untuk *node transmitter* melakukan pengiriman pesan haruslah terjadwal menggunakan protokol TDMA dan waktu harus di sinkronisasi terlebih dahulu agar *transmitter* dan *receiver* memiliki waktu yang sama. Setelah itu melakukan inisialisasi variabel umum yang digunakan untuk semua kode program baik untuk program protokol TDMA, TPSN maupun untuk mengirimkan data. Tabel 5.22 menunjukkan variabel umum yang digunakan pada program *node transmitter*.

Tabel 5.13 Potongan Kode Sumber Variable *Node Transmitter*

	Kode Sumber Variable
1	`Mirf.spi = &MirfHardwareSpi;`
2	`  Mirf.init();`
3	`  Mirf.setRADDR((byte *)"node");`
4	`  Mirf.payload = 32;`
5	`  Mirf.channel = 101;`
6	`  Mirf.config();`
7	`  Serial.println("Waiting...");`
8	`  terdaftar = 0;`
9	`  tersinkron = 0;`
10	`  pinMode(buttonPin, INPUT);`
11	`  tunggu = random(analogRead(A0), 60000);`
12	`  while (tunggu % 17000 < 4000 || tunggu < 28000) {`
13	`    tunggu = random(analogRead(A0), 60000);`
14	`  }`
15	`  x = 0;`
16	`  TDMApacknumb = 0;`
17	`  Sched = 0;`
	`}`

Penggunaan metode protokol TDMA mengharuskan masing-masing *node transmitter* memiliki waktu yang sama dengan *node receiver*. Oleh karena itu perlu dilakukan sinkronisasi waktu antar *node* dengan *receiver* menggunakan protokol TPSN. Pada Protokol TPSN terdapat 2 prosedur utama yaitu prosedur *discovery* prosedur *synchronize.* Prosedur pertama yang dijalankan adalah *discovery*.

Prosedur ini dijalankan untuk melakukan pencarian *node* lain terutama *root* untuk membentuk sebuah topologi. Prosedur lainnya adalah prosedur *synchronize* yaitu prosedur untuk sinkronisasi antar *node* dimana *node* akan saling bertukar data waktu untuk menyetarakan waktunya masing-masing.

Dalam pengimplementasinya, program prosedur *discovery* dibagi menjadi 2 method yaitu *discovery*() dan *discovered*(). Method *discovery*() adalaha method yang berfungsi melakukan *broadcast* paket *discovery* yang berisi waktu penyetaraan menuju semua *node* yang aktif pada saat itu. Sedangkan method *discovered*() adalah method yang melakukan penerimaan paket *discovery* dari *receiver*. Kode program dapat dilihat pada Tabel 5.23.

Tabel 5.14 Potongan Kode Sumber Method discovered() Dan discovery() *Node Transmitter*

Kode Sumber Method *discovered*() Dan *discovery*()	
1	void discovered() {
2	for (int d = 0; d < 4; d++) {
3	AlamatParent[d] = DataTerima[d + 1];
4	}
5	Serial.print("packet discovery received from: ");
6	Serial.println(AlamatParent);
7	Serial.println(DataTerima[5]);
8	Leveltemp[0] = DataTerima[5];
9	Level = strtoul(Leveltemp, NULL, 0);
10	Level = Level + 1;
11	Serial.print("*node* tree level: ");
12	Serial.println(Level);
13	UbahAlamat();
14	Serial.print("my address: ");
15	Serial.println(AlamatSendiri);
16	Mirf.setRADDR((byte *)AlamatSendiri);
17	terdaftar = 1;
18	}
19	void discovery() {
20	sprintf(LevelKirim, "%d", Level);
21	Mirf.setTADDR((byte *)"*node*");
22	strcpy(DataKirim, "B");
23	strcat(DataKirim, AlamatSendiri);
24	strcat(DataKirim, LevelKirim);
25	DataKirim[30] = q;
26	Mirf.send((byte *)&DataKirim);
27	Serial.println("packet discovery sent to all *node*!");
	while (Mirf.isSending()) {

```
28    }
29  }
```

Node transmitter akan mengirimkan suatu paket yang berisi alamat dari tiap *node* dan level dari *node* itu sendiri dengan paket header “B” ke semua *node* aktif yang belum menjadi *root*. Setelah *node* yang dikirimkan paket sudah menerima paket *discovery* tersebut, maka *node* tersebut akan merubah alamatnya dengan method *UbahAlamat*().

Prosedur selanjutnya adalah prosedur *synchronize*. Pada prosedur ini terdapat 2 tahap yaitu *synchronize()* dan *synchronized*(). Method *synchronize*() adalah method yang berfungsi untuk melakukan permintaan sinkronisasi waktu kepada *root*. Sedangkan method *synchronized*() adalah method yang digunakan untuk menerima konfirmasi sinkronisasi waktu. Kode program terdapat pada Tabel 5.24.

Tabel 5.15 Potongan Kode Sumber Method synchronize() dan synchronized()
Node Transmitter

Kode Sumber Method *synchronize*() dan *synchronized*()

```
1  void synchronize() {
2    waktu();
3    Mirf.setTADDR((byte *)AlamatParent);
4    strcpy(DataKirim, "R");
5    strcat(DataKirim, AlamatSendiri);
6    sprintf(TimeStamp1, "%d", milidetik);
7    if (milidetik < 10) {
8      strcat(DataKirim, "0");
9      strcat(DataKirim, "0");
10   } else if (milidetik < 100) {
11     strcat(DataKirim, "0");
12   }
13   strcat(DataKirim, TimeStamp1);
14   sprintf(LevelKirim, "%d", Level);
15   strcat(DataKirim, LevelKirim);
16   Latency = millis();
17   Mirf.send((byte *)&DataKirim);
18   while (Mirf.isSending()) {
19           Serial.println("request synchronization sent!");
20     Serial.println(DataKirim);
21     cetakwaktu();
22     delay(2);
   }
```

```
23  }
24  void synchronized() {
25    Serial.print("begin: ");
26    waktu();
27    cetakwaktu();
28    Serial.println("");
29    waktu();
30    Latency = millis() - Latency;
31    ubahwaktu();
32    waktu();
33    cetakwaktu();
34    Serial.println("synchronized!");
35    Serial.print("delay total: ");
36    Serial.println(Latency);
37    Serial.print("delay propagasi: ");
38    Serial.println(TS4);
39    if (TS4 < 20 && TS4 > 0) {
40      tersinkron = 1;
41    }
      Sched = jadwalpatent + 30;
    }
```

Pada method *synchronize*() paket permintaan sinkronisasi waktu ditandai dengan header "R" menuju ke *receiver*. Paket tersebut berisi alamat *node* pada saat sebelum sinkronisasi dan level *node*. *Node transmitter* akan merubah waktunya setelah menerima paket konfirmasi permintaan sinkronisasi. Setelah mendapatkan paket konfirmasi. Waktu pada *node* akan disamakan dengan waktu yang dikirim dari *root*.

Saat waktu antar *node transmitter* telah setara, maka selanjutnya tiap *node transmitter* dapat melakukan pengiriman data sesuai slot waktu yang telah ditentukan. Slot waktu ini berguna agar tiap-tiap *node* dapat mengirim data secara terjadwal dan dengan waktu tertentu. Slot waktu ditentukan oleh *receiver* agar pembagian slot waktu merata kepada masing-masing *transmitter* sehingga semua *node* mendapatkan jadwal tetap untuk mengirimkan data. Kode program untuk menerima pembagian slot waktu seperti pada Tabel 5.25.

Tabel 5.16 Potongan Kode Sumber Alokasi Waktu *Node Transmitter*

Kode Sumber Alokasi Waktu	
1	`if (!terdaftar && DataTerima[0] == 'B' && DataTerima[1] != 'B') {`
2	`int f = DataTerima[30];`
3	`w = f;`
4	`Serial.print("Jadwal Detik ke : ");`

```
 Serial.println(w);
 discovered();
 if (Level == 2) {
  tunggu = 7000;
    }
  Serial.print("waiting time: ");
  Serial.println(tunggu);

if (detik == w && menit >= 3) {
    TDMAsend();
    jadwal = jadwal + 2;
  if ((jadwal - jadwalpatent) > 3) {
      jadwal = jadwalpatent;
     }

```

Bagian method terakhir pada *node transmitter* adalah method `TDMAsend()` yang berfungsi untuk pengiriman data dari sensor yang terpasang pada masing-masing *node*. Kode program method ini dapat dilihat pada Tabel 5.26.

Tabel 5.17 Potongan Kode Sumber TDMAsend() Node Transmitter

Kode Sumber *TDMAsend*()	
1	`void TDMAsend() {`
2	`  TDMApacknumb = TDMApacknumb + 1;`
3	`  TDMApacknumb = TDMApacknumb % 10;`
4	`  sprintf(TDMApacknumb2, "%d", TDMApacknumb);`
5	`  float suhu = dht.readTemperature();`
6	`  float data = suhu;`
7	`  Mirf.setTADDR((byte *)"1111");`
8	`  strcpy(DataKirim, "T");`
9	`  strcat(DataKirim, AlamatSendiri);`
10	`  strcat(DataKirim, TDMApacknumb2);`
11	`  strcat(DataKirim, "0");`
12	`  sprintf(LevelKirim, "%d", Level);`
13	`  strcat(DataKirim, LevelKirim);`
14	`  Mirf.setTADDR((byte *)"1111");`
15	`  DataKirim[31] = data;`
16	`  Mirf.send((byte *)&DataKirim);`
17	`  while (Mirf.isSending());`
18	`  Serial.print("temperature");`
	`  Serial.print(" : ");`

```
19    Serial.print(suhu);
20    Serial.println("oC");
21    Serial.print("from level : ");
22    Serial.println(Level);
23    cetakwaktu();
24    Serial.println("data sensor sent!");
25  }
```

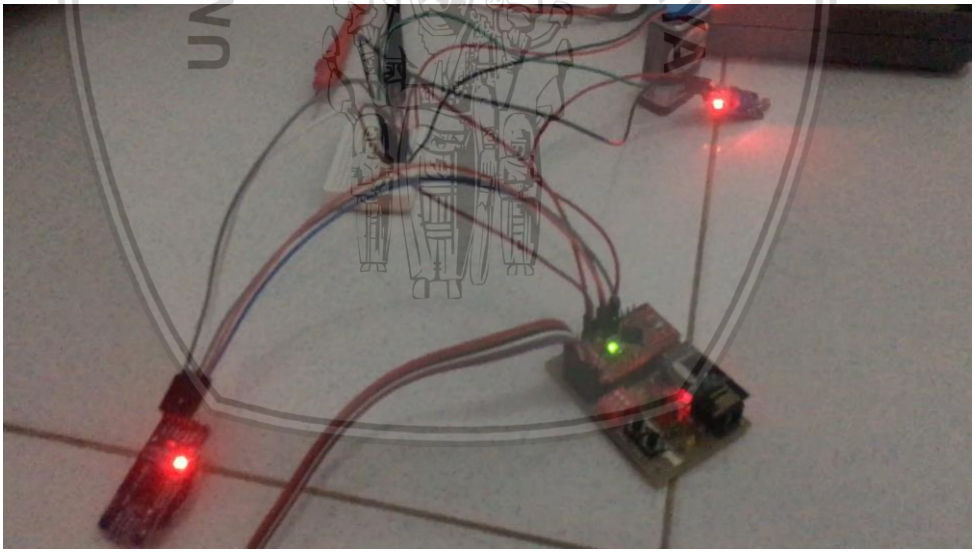
BAB 6 PENGUJIAN DAN ANALISIS

Dalam bab ini akan menjelaskan tentang skenario pengujian, pengujian alat serta analisis dari penelitian.

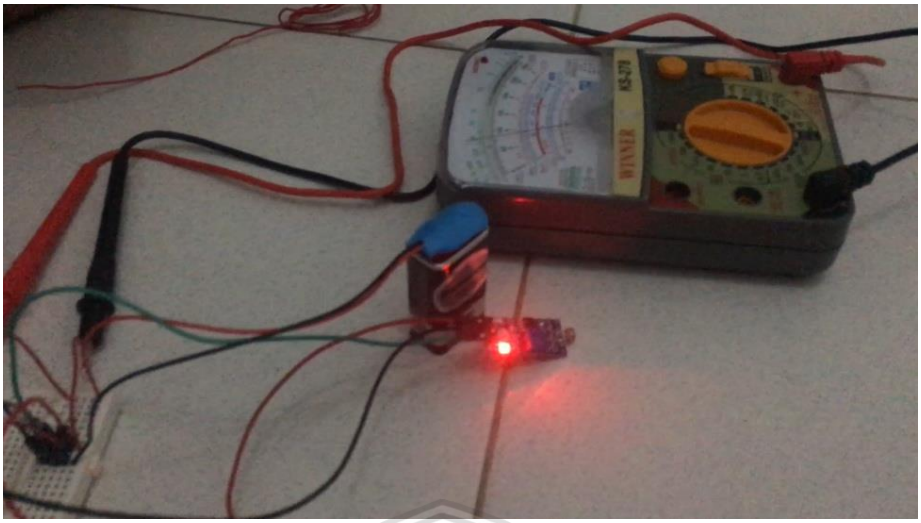
6.1 Skenario Pengujian

Pengujian yang dilakukan dalam penelitian ini adalah bertujuan untuk menguji fungsionalitas sistem serta untuk mengukur besaran arus pada masing-masing *node transmitter* yang sudah terhubung dengan sumber daya baterai 9 Volt. Pada tiap *node* terdapat sensor-sensor yang berbeda seperti DHT11, LDR, Rain Drop dan Soil Moisture.

Skenario pengujian ini akan terbagi menjadi 2 yaitu pengujian fungsional dimana akan diuji fungsionalitas *transmitter* maupun *receiver* dan pengujian untuk arus *low power*.Pada tiap pengujian menggunakan 4 *node transmitter* dan 1 *node receiver*. Pada pengujian arus akan diukur dengan multimeter yang dihubungkan pada VCC dan ground pada mikrokontroler agar dapat melihat penggunaan arusnya seperti pada Gambar 6.1 dan 6.2. Multimeter yang digunakan dalam penilitian ini adalah multimeter analog.



Gambar 6.1 *Node Transmitter* Pada Skenario Pengujian



Gambar 6.2 Pengukuran Arus pada *Node* Transmitter

6.2 Pengujian dan Analisis fungsional pada Rangkaian Node *Transmitter* dan *Receiver*

Pada node transmitter, masing-masing node berfungsi untuk mengirim hasil dari sensing data dari tiap-tiap sensor yang ada. Selain itu juga melakukan komunikasi dengan *node receiver* dimana *node receiver* akan menerima hasil dari *node transmitter* yang akan ditampilkan melalui *serial monitor.* Fungsi *sleep* dan *wake up interrupt* pada node transmitter juga akan diuji pada pengujian ini.

6.2.1 Tujuan

Pengujian fungsional bertujuan agar mengetahui jika seluruh fungsi-fungsi yang berjalan pada tiap node *transmitter* maupun *receiver* berjalan sesuai dengan tujuannya masing-masing.

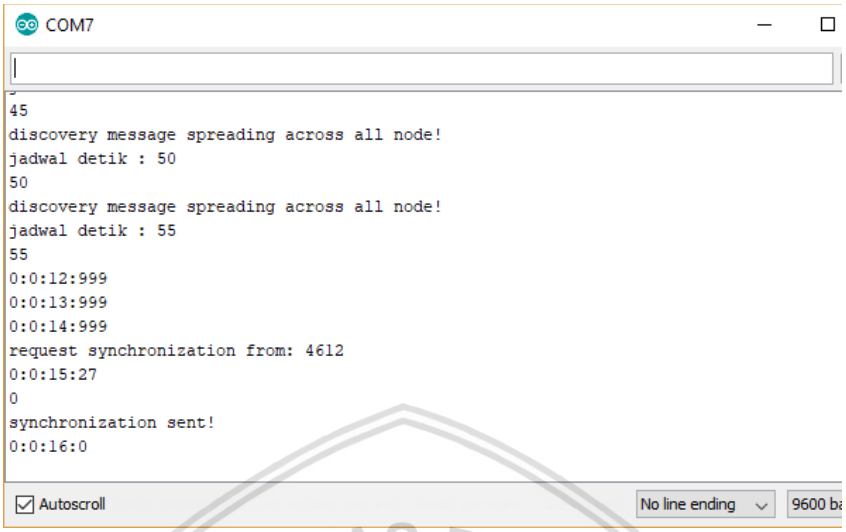
6.2.2 Langkah-Langkah Pengujian

Pada pengujian ini yang pertama akan dilakukan upload kode program pada *node receiver* dan kemudian pada *node transmitter*. Setelah itu masing-masing node akan saling berkomunikasi dengan mengirim *message discovery* yang berisi waktu penjadwalan untuk tiap *node transmitter* melakukan pengiriman ke *node receiver.* Setelah mendapatkan waktu penjadwalan, maka tiap-tiap node akan mengirim hasil data sensor yang kemudian akan di tampilkan melalui *serial monitor* yang sudah diproses melalui *node receiver*.

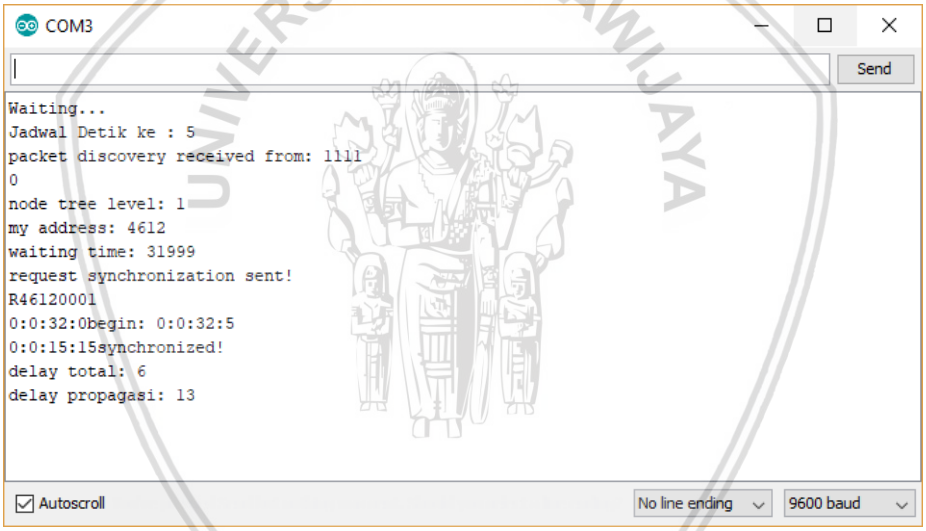
6.2.3 Hasil

Pada Gambar 6.3 node receiver akan melakukan *broadcast* ke semua *node transmitter* yang berisi pesan *discovery* yang akan diterima oleh semua *node*.Setelah menerima pesan *discovery* maka antara node receiver dan transmitter akan melakukan pengiriman pesan *synchronize*. Seperti pada Gambar 6.4 dapat dilihat saat antara node receiver dan node transmitter telah berhasil

melakukan inisialisasi pesan *synchronized* maka akan di dapat waktu penjadwalan, alamat dan waktu *delay*nya.

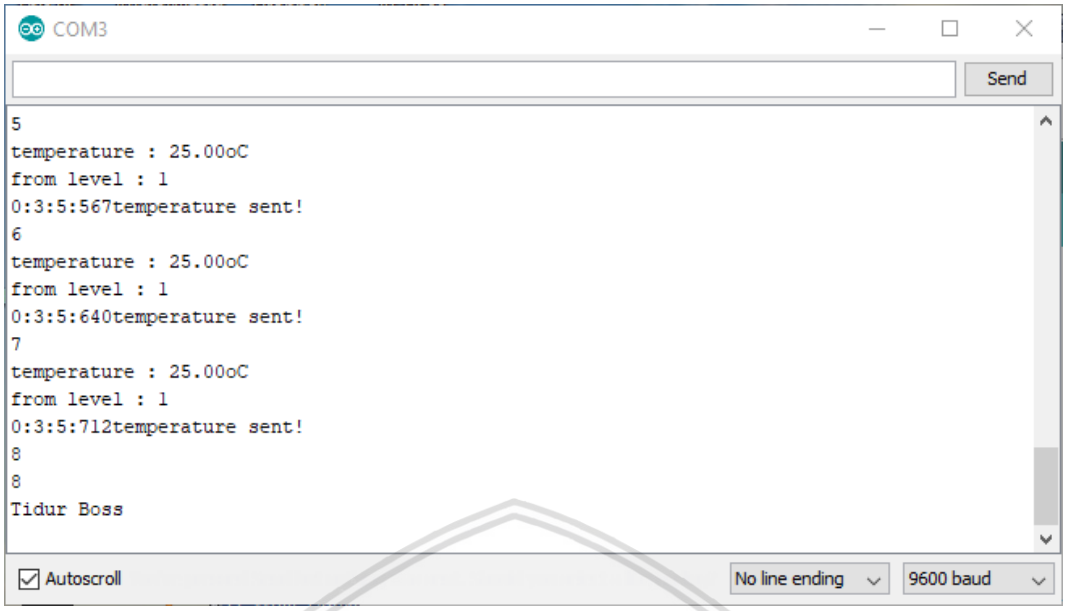


Gambar 6.3 *Node Receiver* Saat Menjalankan Fase Discovery

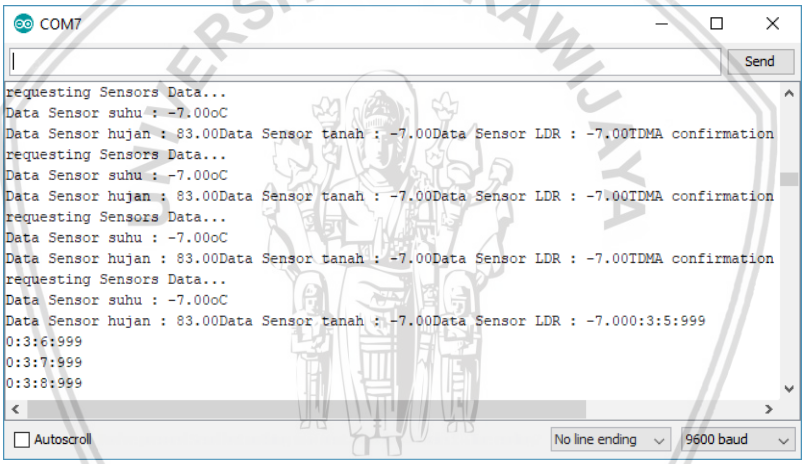


Gambar 6.4 *Node Transmitter* Saat Memasuki *Fase Synchronized*

Setelah memasuki fase *synchronized,* node transmitter akan mengirimkan data sensornya seperti yang terlihat pada Gambar 6.5 dan kemudian akan memasuki *sleep mode* dimana node sensor tersebut akan bangun saat menerima *interrupt* dari RTC dengan waktu yang ditentukan di dalam kode program.



Gambar 6.5 *Node Transmitter* Saat Memasuki *Sleep Mode*



Gambar 6.6 *Node Receiver* Saat Menerima Hasil dari Sensor

Pada Gambar 6.6 dapat dilihat bahwa *node receiver* telah berhasil menerima data sensor yang dikirim oleh *node transmitter*. Data sensor merupakan hasil dari *analog* atau *digital read* dari masing-masing sensor yang ada di *node transmitter*. Namun, *node receiver* akan terus melakukan *listening* meskipun *node-node transmitter* telah memasuki *low power mode*.

Setelah melakukan pengiriman *node transmitter* akan memasuki sleep mode. *Node – node* tersebut akan kembali aktif jika menerima sinyal *interrupt* yang berasal dari RTC. Pada Gambar 6.7 dapat dilihat bahwa node transmitter telah bangun dari *sleep mode* sesuai dengan waktu yang diatur di dalam kode program.

Gambar 6.7 *Node Transmitter* Saat Menerima *Wake Up Interrupt*

6.2.4 Analisis

Dari pengujian yang telah dilakukan, dapat disimpulkan bahwa komunikasi antar node transmitter dan receiver berlangsung dengan baik dimana *node transmitter* mampu mengirimkan data sensornya dan *node receiver* dapat menampikannya di *serial monitor*. Selain itu, *interrupt* dari RTC mampu membangunkan *node transmitter* dari *sleep mode* sesuai dengan jadwal yang telah ditentukan.

6.3 Pengujian dan Analisis *Low Power* pada Rangkaian *Node Transmitter*

Low power mode pada penelitian ini berfungsi untuk mengatur sistem agar dapat bekerja dengan daya yang rendah saat telah menerima *intterupt* dari RTC yang waktunya ditentukan melalui Arduino IDE dalam kode program. Jika pada penelitian sebelumnya telah menggunakan *library* Jeelib yang berguna untuk menekan penggunaan arus pada *node transmitter*, maka pada penelitian ini tidak akan menggunakan library tersebut karena telah penggunaan modul RTC yang mampu memberikan *interrupt* agar sistem dapat memasuki *mode sleep.* Dalam pengujian ini, yang akan diukur hanyalah penggunaan arus pada *node transmitter* yang telah dipasangi modul RTC dimana mekanisme *low power* ini diimplementasikan.

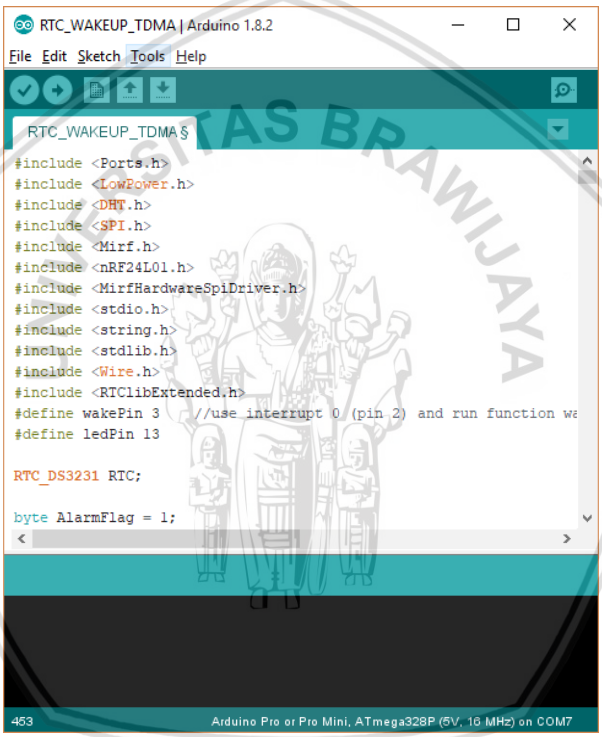
6.3.1 Tujuan

Tujuan dari pengujian ini adalah untuk mengetahui perbedaan besaran arus pada masing-masing *node* yang telah terhubung dengan baterai 9 Volt pada saat tidak *sleep* dan saat *sleep* serta efektifitas mekanisme *low power* pada sistem ini. Setelah mendapatkan hasil pengujian arus, maka akan dihitung persentasi rata-rata konsumsi penggunaanya.

6.3.2 Langkah-Langkah Pengujian

Pada pengujian ini akan diujikan dengan tiap *node* transmitter dan *node* receiver yang saling berkomunikasi satu dengan yang lain. Pada saat mengirimkan data dan *idle node* diasumsikan sebagai *state* tidak *sleep* sedangkan pada saat *node* telah menerima *interrupt* maka diasumsikan sebagai *state sleep.*

Dalam pengujian ini masing-masing *node* receiver maupun transmitter akan dimasukan kode program seperti pada Gambar 6.6 untuk menjalankan fungsinya masing-masing dari Arduino IDE. Kemudian *node* akan melakukan tugasnya masing-masing hingga *node* transmitter memasuki *mode sleep.* Pada tiap *state* Arduino konsumsi arus akan diukur menggunakan multimeter. Setelah selesai mendapatkan data maka akan dilakukan perhitungan selisih dan persentase rata-rata penggunaan arusnya.



Gambar 6.8 Proses *Upload* Program ke *Node*

6.3.3 Hasil

Dalam penelitian ini didapati hasil pengukuran arus saat *node transmitter* dalam *low power mode* dan *non-low power mode. Low power mode* adalah *state* dimana *node transmitter* telah menerima *interrupt* dari RTC. Sedangkan *non-low power mode* adalah state dimana *node transmitter* masih melakukan sensing data sensor dan komunikasi dengan *receiver*. Hasil pengukuran arus dapat dilihat pada Tabel 6.1

Tabel 6.1 Hasil Pengujian Konsumsi Arus

Node Sensor	Low Power Mode (mA)	Non-Low Power Mode (mA)	Selisih Arus (mA)
DHT11	35	60	25
Soil Moisture	45	75	30
LDR	40	80	40
Rain Drop	40	75	35
Rata-rata	40	72,5	32,5

Dari Tabel 6.1 didapati hasil rata-rata penggunaan arus low power sebesar 40 mA dan non-low power sebesar 72,5 mA. Selain itu didapati rata-rata selisih arus sebesar 32,5 mA.

6.3.4 Analisis

Setelah mendapatkan hasil penelitian dari skenario pengujian, maka selanjutnya akan dilakukan perhitungan persentase terhadap konsumsi arus *low power* dan *non-low power*. Untuk menghitung persentasi dari penggunaan arus dapat menggunakan rumus sebagai berikut :

$$\% = \frac{\bar{x} - \bar{y}}{\bar{x}} \text{ X } 100$$

Keterangan :

% adalah persentase penghematan konsumsi arus

$\bar{x}$ adalah rata-rata konsumsi arus *low power* dalam *milliampere*

$\bar{y}$ adalah rata-rata konsumsi arus *non-low power* dalam *milliampere*

Berdasarkan data hasil pengujian dengan menggunakan rumus diatas maka didapati bahwa persentase total penghematan penggunaan arus pada seluruh *node* transmitter adalah hingga 81,25 % saat mekanisme *low power* berjalan dengan pada *node transmitter*.

Dengan penggunaan rancangan perangkat keras yang hampir sama dengan penelitian sebelumnya, maka dapat dianalisis bahwa dengan penggunaan metode *wake up interrupt* modul RTC pada sistem mampu menurukan penggunaan arus yang cukup signifikan. Dari segi program pada sistem yang menjalankan protokol TDMA dan TPSN, didapati bahwa dengan adanya mode *sleep* tidak banyak mempengaruhi jalannya program. Sehingga dapat disimpulkan bahwa integrasi antara mekanisme *low power* pada sistem dan sinkronisasi waktu serta

penjadwalan pengiriman telah berhasil tanpa banyak menggangu jalannya sistem secara keseluruhan.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil penelitian mulai dari tahap perancangan, implementasi hingga pengujian dan analisis hasil pengujian yang telah selesai dilaksanakan, maka penulis dapat mengambil beberapa kesimpulan sebagai berikut :

1. Semua *node receiver* maupun *transmitter* mampu berkomunikasi satu dengan yang lain serta menjalankan fungsinya masing-masing.
2. Penggunaan mekanisme *low power* pada *node transmitter* mampu menghemat sebesar 81,25 % dari mA arus penggunaan tanpa mekanisme *low power.*
3. Penerapan metode *wake up interrupt* dari modul RTC yang diintegrasikan *low power mode* dengan rancangan perangkat keras yang sedemikian rupa dapat berintergrasi dengan baik. Dengan demikian tujuan utama penelitian ini telah tercapai.

7.2 Saran

Berdasarkan kesimpulan yang telah dibuat oleh penulis, maka penulis memiliki beberapa saran untuk penelitian selanjutnya yang akan mengembangkan penelitian ini yaitu :

1. Dapat mengembangkan desain interaksi untuk sistem ini agar memudahkan pengguna untuk melakukan pengaturan pada input waktu maupun interaksi secara keseluruhan dengan sistem.
2. Mematikan fungsi yang tidak digunakan pada mikrokontroler seperti lampu LED serta modul lainnya agar menghemat penggunaan daya pada sistem lebih banyak lagi.
3. Mampu menerapkan metode penjadwalan lain seperti FDMA (Frequency Division Multiple Access) atau CDMA (Code Division Multiple Access).

DAFTAR PUSTAKA

Adafruit. (2017). *Adafruit FT232H Breakout.* Retrieved March 2018, from Adafruit Company : https://learn.adafruit.com/adafruit-ft232h-breakout/serial-uart

Adhnaufal, A. F. (2017). *IMPLEMENTASI LOW POWER WIRELESS SENSOR NETWORK UNTUK PENGUKURAN SUHU BERBASIS NRF DENGAN PENJADWALAN PENGIRIMAN DATA*. Malang, Indonesia: Universitas Brawijaya.

Arduino. (2016). *Product Arduino*. Retrieved April 2018, from Arduino Company : http://www.arduino.cc/en/Main/Products

ASA Nordic Semiconductor. (2016). *Nordic Semiconductor NRF24L01 Product Specification*. Retrieved September 2016, from Nordic Semiconductor : http://nordicsemi.com/eng/nordic/download_resource/8041/1/62435711

Elson, J. (n.d.). *Time Synchronization for Wireless Sensor Networks.* Los Angeles: University of California.

Erwanda, A. N. (2016). *IMPLEMENTASI TIME SYNCHRONIZATION PADA WSN UNTUK METODE TDMA MENGGUNAKAN ALGORITMA TPSN.* Malang, Indonesia: Universitas Brawijaya.

Ganeriwal, Saurabh. Et al,. (2003). *Timing-sync Protocol for Sensor Network*. Los Angeles : University of California.

Hidayat, M. F. (2018). Implementasi Low Power Multi Sensor *Node* pada Wireless Sensor Network. Malang, Indonesia : Universitas Brawijaya.

Kadir, A., 2013. *Panduan Praktis Mempelajari Aplikasi Mikrokontroler dan Pemrogramannya menggunakan Arduino.*. Yogyakarta: Andi Komputindo.

Prasojo, Gatut. (2016). *IMPLEMENTASI LOW POWER MODE PADA WIRELESS SENSOR NODE*. Malang, Indonesia : Universitas Brawijaya.

SparkFun. (2017). *SparkFun USB to serial UART Boards Hookup Guide*. Retrieved April 2018, from Sparkfun : https://learn.sparkfun.com/tutorials/sparkfun-usb-to-serial-uart-boards-hookup-guide?_ga=1.47589859.512431245.1485075572

SparkFun. (2017). *Reducing Arduino Power Consumption*. Retrieved March 2018, from SparkFun : https://learn.sparkfun.com/tutorials/reducing-arduino-power-consumption

SparkFun. (2017). *I2C*. Retrieved May 2018, from SparkFun : https:// https://learn.sparkfun.com/tutorials/i2c

Zhou, Y., Zhou, Q., Kong, Q. & Cai, W., 2012. Wireless Temprature & Humidity Monitor and Control System.